Defence Research and Development Canada
Recherche et développement pour la défense Canada

DEFENCE R&D DÉFENSE

Security classification using automated learning (SCALE)

Optimizing statistical natural language processing techniques to assign security labels to unstructured text

J. David Brown and Daniel Charlebois

Defence R&D Canada – Ottawa

Canada

Technical Memorandum
DRDC Ottawa TM 2010-215
December 2010

Security classification using automated learning (SCALE)

Optimizing statistical natural language processing techniques to assign security labels to unstructured text

J. David Brown
Defence R&D Canada – Ottawa

Daniel Charlebois
Defence R&D Canada – Ottawa

Defence R&D Canada – Ottawa
Technical Memorandum
DRDC Ottawa TM 2010-215
December 2010

Principal Author

Original signed by J. David Brown

J. David Brown

Approved by

Original signed by Julie Lefebvre

Julie Lefebvre
Head/NIO Section

Approved for release by

Original signed by Chris McMillan

Chris McMillan
Head/Document Review Panel

Abstract

Automating the process of assigning security classifications to unstructured text would facilitate a transition to a data-centric architecture—one that promotes information sharing, in which all data in an organization are electronically labelled. In this document, we report the results of a series of experiments conducted to investigate the effectiveness of using statistical natural language processing and machine learning techniques to automatically assign security classifications to documents. We present guidelines for selecting parameters to maximize the accuracy of a machine learning algorithm's classification decisions for several well-defined collections of documents. We examine the significance of a document's topic and the effect of security policy changes on the ability of our system to automate classification; we include design recommendations to address both topic and policy considerations. Our classification techniques prove effective at assessing a document's sensitivity, achieving accuracies upwards of 80%.

Résumé

Le fait d'automatiser le processus d'attribution des classifications de sécurité aux textes non structurés faciliterait l'adoption d'une architecture axée sur les données où l'on favoriserait l'échange de renseignements et où toutes les données d'une organisation seraient étiquetées électroniquement. Dans le présent document, nous présentons les résultats obtenus à la suite d'une série d'expériences visant à évaluer l'efficacité des techniques de traitement du langage naturel statistique et d'apprentissage automatique pour attribuer automatiquement une classification de sécurité à un document. Nous précisons des directives quant au choix de paramètres qui permettront de maximiser le bien-fondé des décisions de l'algorithme d'apprentissage automatique quant à la classification de plusieurs documents bien définis. Nous évaluons l'importance du sujet traité dans un document et l'incidence des modifications aux politiques de sécurité sur la capacité du système d'automatiser l'attribution d'une classification. Nous proposons des recommandations de conception qui permettront de tenir compte des divers facteurs à considérer quant au sujet et aux politiques. Nos techniques de classification se sont révélées efficaces au moment d'évaluer la sensibilité d'un document en affichant un taux d'exactitude de plus de 80 p. 100.

This page intentionally left blank.

Executive summary

Security classification using automated learning (SCALE)

J. David Brown, Daniel Charlebois; DRDC Ottawa TM 2010-215; Defence R&D Canada – Ottawa; December 2010.

Background: NATO allies are increasingly demanding the ability to securely and efficiently share information among international partners and within national organizations and agencies [1]. Ultimately, to mediate access to data of different sensitivities and to move data between different security domains, all data must be securely labelled with the appropriate classification. A majority of the data found in defence networks is unstructured and unlabelled, with large quantities of new data generated daily. Currently, assigning security labels to new and existing text data requires a trained evaluator to read and understand the text, and apply the appropriate classification based on experience and security policy. To efficiently label all new and existing data in networks belonging to the Department of National Defence (DND), we require a reliable method to automate (or partially automate) the assignment of security classifications. Not only would an automated security classification system help to make the current manual process less labour-intensive, it would also ensure a more consistent and unbiased application of security policy and would significantly reduce the problem of overclassification that plagues many defence networks [2, 3]. Unfortunately, while automatically categorizing data according to topic has been extensively studied, there is little published research focusing on how to automatically assess a document's sensitivity.

In this paper, we investigate the effectiveness of statistical natural language processing (SNLP) and machine learning in assessing the sensitivity of unstructured text. We conducted a series of experiments to determine how best to apply existing SNLP and machine learning techniques to the task of assigning a security classification to a document, optimizing the parameters of our system by testing it on a collection of hundreds of declassified government documents obtained from the Digital National Security Archive (DNSA) [4].

Principal results: Using a traditional machine learning approach, we achieved classification accuracies near 80%. A more sophisticated approach (where documents were left unlabelled if our algorithms failed to arrive at a satisfactory decision) allowed us to achieve accuracies higher than 90% on a subset of the documents. In addition to evaluating the classification accuracies obtained using a number of approaches, we showed that the classifier was highly sensitive to a document's topic and

to security policy changes. Based on our results, we proposed an optimized design for a classification system based on SNLP, including methods to account for topic sensitivity.

Significance of results: Our results demonstrate that statistical natural language processing with machine learning is effective in identifying the sensitivity of unstructured text, assuming that a sufficient number of correctly labelled relevant training documents are available. To our knowledge, this is the first paper in the scientific literature to publish results documenting the accuracy of using machine learning to determine security classification. These results represent an important step towards the production of a robust automated classification system, clearly indicating that SNLP should play a major role in a broader automated system.

Future work: While this work demonstrates the effectiveness of applying SNLP and machine learning to the task of automated classification, there remain many open research questions on the road to developing a robust automated system that could be used by DND. Ultimately, we believe that SNLP should form an integral part of a larger automated system that also takes into account pre-existing contextual knowledge about a document and its associated security policy; how best to incorporate this context with machine learning techniques is unclear and is a topic we are actively studying. Additional topics that we intend to explore are methods to adapt an automated system to changes in policy, methods to identify relevant high-level topical domains, and the development of a design philosophy that describes how an automated system could be integrated with existing information technology architectures.

Sommaire

Security classification using automated learning (SCALE)

J. David Brown, Daniel Charlebois ; DRDC Ottawa TM 2010-215 ; R & D pour la défense Canada – Ottawa ; décembre 2010.

Contexte : Les alliés de l'OTAN exigent de plus en plus un échange de renseignements sécuritaire et efficace entre les partenaires internationaux et au sein des divers organismes nationaux [1]. En définitive, pour faciliter l'accès à des données ayant des sensibilités différentes et déplacer des données entre différents domaines de sécurité, il est essentiel d'étiqueter toutes les données de façon sécuritaire et d'attribuer la bonne classification. La plupart des données disponibles dans les divers réseaux de la Défense, dans lesquels plusieurs nouvelles données sont générées chaque jour, ne sont ni structurées ni étiquetées. À l'heure actuelle, l'attribution d'une étiquette de sécurité aux données d'un texte n'est possible que si un évaluateur ayant reçu une formation à cet effet a lu et compris le texte en question avant d'attribuer la bonne classification en fonction de son expérience et des politiques de sécurité. Pour étiqueter de manière efficace toutes les données disponibles dans les réseaux du ministère de la Défense nationale (MDN), nous avons besoin d'un processus fiable pour automatiser, en totalité ou en partie, l'attribution d'une classification de sécurité. Un système automatisé d'attribution de classification de sécurité permettrait non seulement de rendre le processus manuel actuel un peu moins exigeant sur le plan de la main-d'œuvre, mais aussi d'assurer une mise en application uniforme et impartiale des politiques de sécurité, ce qui réglerait en grande partie le problème de surclassification qui touche bon nombre de réseaux de la Défense [2, 3]. Malheureusement, bien que l'on ait étudié en profondeur la catégorisation automatique de données en fonction du sujet, peu de recherches axées sur l'évaluation automatique de la sensibilité d'un document ont été publiées.

Dans le présent document, nous examinons l'efficacité du traitement du langage naturel statistique (TLNS) et de l'apprentissage automatique au moment d'évaluer la sensibilité d'un texte non structuré. Nous avons effectué une série d'expériences pour déterminer la meilleure façon d'appliquer les techniques de TLNS et d'apprentissage automatique afin d'attribuer une classification de sécurité à un document et d'optimiser les paramètres de notre système en les essayant sur des centaines de documents gouvernementaux déclassifiés, obtenus grce aux archives numériques sur la sécurité nationale [4].

Principaux résultats : Lorsque nous avons utilisé une approche d'apprentissage automatique traditionnelle, le taux d'exactitude des classifications était d'environ 80

p. 100. Une approche plus sophistiquée, selon laquelle les documents demeuraient sans étiquette lorsque la décision de nos algorithmes était insatisfaisante, nous a permis d'obtenir un taux d'exactitude de plus de 90 p. 100 suite à l'attribution d'une classification à un sous-ensemble de documents. En plus d'évaluer l'exactitude de la classification obtenue à l'aide d'un certain nombre d'approches, nous avons conclu que le classificateur était très sensible au sujet du document et aux modifications aux politiques de sécurité. En tenant compte des résultats obtenus, nous avons proposé un système de classification optimisé fondé sur le TLNS, notamment les processus d'évaluation de la sensibilité du sujet.

Importance des résultats : Les résultats démontrent que le traitement du langage naturel statistique combiné à l'apprentissage automatique est un moyen efficace qui permet d'identifier la sensibilité d'un texte non structuré lorsqu'il y a un nombre suffisant de documents de formation pertinents étiquetés correctement. À notre connaissance, il s'agit du premier article scientifique où l'on présente des résultats documentant l'exactitude des techniques d'apprentissage automatique au moment d'attribuer une classification de sécurité. Ces résultats constituent une étape importante en vue de la création d'un système efficace de classification automatisée, car ils indiquent clairement que l'on devrait accorder une plus grande importance au TLNS dans un système automatisé plus vaste.

Prochaines étapes : Bien que le présent document prouve que l'utilisation des techniques de TLNS et d'apprentissage automatique pour attribuer une classification automatisée est efficace, on doit encore répondre à de nombreuses questions de recherche avant de créer un système automatisé solide que le MDN pourrait utiliser. En définitive, nous sommes d'avis que le TLNS devrait faire partie intégrante d'un système automatisé plus vaste qui tiendrait également compte des connaissances contextuelles actuelles sur un document et des politiques de sécurité connexes. Nous ignorons quelle serait la meilleure façon d'intégrer ce contexte aux techniques d'apprentissage, mais c'est un sujet que nous étudions activement. La façon d'adapter un système automatisé à l'évolution des politiques, la manière d'identifier les domaines spécialisés pertinents de haut niveau et l'élaboration d'une philosophie de conception décrivant comment on pourrait intégrer un système automatisé à l'architecture de la technologie de l'information existante font partie des autres sujets que nous souhaitons approfondir.

Table of contents

Acknowledgements

We would like to thank Dr. Kathryn Perrett for her invaluable suggestions regarding both the analysis and presentation of our results.

1 Introduction: The need for automated security classification

Throughout the past decade, NATO allies have identified the need for a fundamental shift in the culture of information management and security: away from the current philosophy of "need to know" and towards a philosophy of "need to share" [1, 5, 6]. Need-to-know architectures are typified by network silos that separate caveats and classifications by air gaps, whereas need-to-share architectures focus on network convergence and on the exploitation of intelligent cross-domain data guards. The need for secure information sharing among federal agencies and between allies has been recognized as a necessary step towards gaining information superiority, ultimately ensuring that all allies can access the most up-to-date and relevant information, while protecting it from adversaries [7].

Critical to achieving the goal of secure information sharing is the requirement that *all* information objects be *digitally* labelled with sufficient metadata to support the application of access management systems [8]. Access to information is mediated based on the object's metadata, the applicable security policy, and the user's identity and attributes. Whether access control is enforced using cross-domain guards or through a data-centric mechanism such as in [9], trusted metadata is a basic system requirement. A data element's security classification and community of interest (COI) are required metadata for enforcing information sharing policies. Unfortunately, while the Government of Canada Security Policy (GSP) [10] and Department of National Defence (DND) Metadata Application Profile [11] both clearly articulate the need for government data to be accurately marked with the appropriate security classification, they provide little concrete guidance on determining the appropriate classification for an unlabelled document.

Frequently, the task of assigning a security classification to a document rests in the hands of the information owner (i.e., the originator of the document) or a security officer. Not only can this task be time-consuming, it can result in inconsistent classifications—to some degree it is a subjective exercise that depends upon the knowledge and training of the individual assessing a document's sensitivity. In addition, there is a well-documented tendency for individuals to err on the side of caution and overclassify documents; in fact, recent expert testimony before the U.S. Congress reported an estimated fifty percent of military-related U.S. government documents were overclassified [2]. Maj. Gen. Flynn—the top U.S. military intelligence officer in Afghanistan—recently reported that overclassification is hindering the intelligence effort in Afghanistan [3]. Inconsistent classification and overclassification both hamper efforts to efficiently share information, not to mention the additional resources devoted to protecting overclassified documents. Although overclassification is a larger systemic issue, there is nevertheless always a risk of under-classification as well, which

leaves open the possibility of sensitive information falling into the wrong hands.

The ability to reliably automate the process of security classification would increase organizational efficiency, ensure the consistent application of security labels, and facilitate the transition to a data-centric infrastructure that supports the imperative of "need to share". Unfortunately, the study of automated security classification is relatively unexplored in the open literature. Digital labelling solutions abound (e.g., see [12]) but these rely on a human operator to assign a label. Most content scanning solutions rely on the availability of black-lists and/or white-lists of words, which flag data as sensitive.

This paper presents the results of an extensive series of experiments investigating the application of existing *statistical text categorization* methods to the problem of automating the security classification of unstructured text, following on from work originally proposed in [13]. Traditional machine learning methods are tested and optimized to determine which ones are best suited to categorize documents based on sensitivity. Also investigated is the ability of a trained machine learner to adapt to documents from different topic domains and from different eras (intended to reflect a change in government policy). To our knowledge, this is the first paper in the scientific literature to report any results of the performance of machine learning algorithms used for security classification. Ultimately, this paper demonstrates the extent to which machine learning (based purely on the statistical analysis of text) can be relied upon as a tool to assess the sensitivity of a document, identifying those instances in which it succeeds and also those in which it fails.

We begin in Section 2 by discussing current research and technology related to automated security classification, including a brief discussion of some commercially available content scanning products. In Section 3, an automated security classification system model is presented that forms the basis for all the experiments described in the remainder of the paper; existing machine learning techniques exploited by the classification system are also briefly described. Section 4 describes the experiments that were conducted and reports the results, providing a comprehensive discussion of the observations and explaining their importance. General recommendations and suggestions for future work are presented in Section 5. Finally, Section 6 summarizes the paper and presents conclusions.

2 Existing research and related technology

While our focus in this paper is on evaluating and optimizing machine learning techniques for the purposes of security classification, it is important to note that the use of machine learning for topical sort and categorization is well understood and is exploited in many commercial products. This section briefly discusses pertinent research and technology that could be applied to automated security classification, including commercially available content analyzers and cross-domain guards used in military applications [1].

2.1 Existing research in automated document security classification

There is very little in the open literature that specifically addresses the problem of automating the security classification of unstructured text. Discoveries and techniques from the fields of information retrieval and text categorization provide many of the necessary tools to investigate the security classification problem, but do not tackle it head on. In [13], we discussed how most research in document categorization focuses on *topical* classification—i.e., identifying the topic of a document—whereas security classification is non-topical, making it a more challenging problem. We identified some limited work done by others in this field (e.g., [14, 15]) who also observed the lack of published data. In addition, in [13], we provided a thorough description of state-of-the-art machine learning classification techniques for both topical and non-topical scenarios and identified the need for experimental work investigating the application of such techniques to security classification. For a complete discussion of related research, we recommend [13].

2.2 Commercial content scanning technology

In [16], a companion piece to this paper, Magar completed a thorough review of commercially available products containing content analysis capability. From the fields of anti-spam software, Data Leakage Prevention (DLP) suites, and content management solutions, the providers of enterprise content management software were found to have the most sophisticated content analysis capabilities. These packages have a myriad of applications, among which is the ability to categorize data using machine learning techniques (although the precise implementation is unclear and is guarded as intellectual property). The vendors do not provide statistics on their

1. Cross-domain guards are typically used to mediate data flow between networks of different sensitivities and are tasked with ensuring that sensitive data are not released onto networks lacking the requisite level of security control.

products' categorization accuracy and there are no publicly available reports on the abilities of these products to analyze a document's sensitivity. The account in [16] is the first work we are aware of that analyzes the use of a commercial off-the-shelf (COTS) content analysis engine to perform security classification.

2.3 Cross-domain guard technology

The Unified Cross Domain Management Office (UCDMO) [2] has compiled a list of officially-approved cross-domain information transfer solutions, including the popular Radiant Mercury and Information Support Server Environment (ISSE) guards, the Data Sync Guard made by BAE, Northrop Grumman's SMART.neXt guard, and Boeing's eXMeritus HardwareWall. These guards are high-assurance products intended to connect multiple networks operating at different levels of classifications and with different security policies. By design, the guards allow information transfer between two networks only if the information meets strict formatting requirements and the content has been appropriately sanitized.

The Radiant Mercury and ISSE guards are most effective on highly structured data that are amenable to the implementation of simple rules. The guards parse incoming data into a set of pre-selected attributes and apply rule sets to ensure that none of the data violates any of the operator-defined constraints. In general, data not conforming to a specified format will be quarantined for human operator review [17].

As described in [18], the Data Sync Guard checks documents for so-called "dirty" and "clean" words, scans for user-applied labels throughout the document's content, and ensures that the document conforms to one of a set of supported XML schemas. It also supports the search of meta-data to ensure that pre-defined rules are met.

Northrop Grumman's SMART.neXt guard refers to their policies as Direction Based Access Control (DBAC), whereby any checks performed by the guard are configurable based on the direction of data flow. Typical DBAC policies are dirty word search, security label scanning, and attachment filtering to block, strip, or force a review of non-formatted attachments. Company literature recommends the use of third-party software for potentially automating the review of attachments [19].

Similar to the other guards, the eXMeritus HardwareWall reviews internal classification tags, supports regular expression (dirty word) searching, reviews metadata fields, and ensures compliance with recognized schemas. Data are also subject to signature review to ensure their integrity [20].

All the guards support at least some form of dirty word search, perform checks to

2. The UCDMO was founded in 2006 and co-ordinates all U.S. Department of Defence (DoD) activities related to cross-domain information sharing.

ensure data is formatted according to acceptable specifications, and examine user-defined metadata and/or labels. To maintain a high level of assurance, the guards typically limit content scanning functionality to these basic requirements and there is little to distinguish them from one another in this regard. To employ more advanced content scanning techniques in conjunction with existing guards (i.e., guards that have been certified and accredited) it would not be feasible to make radical changes to the guards themselves, which would require repeating the certification and accreditation process on the guard. A reasonable alternative would be to conduct content scanning in an off-board module and deliver the output to the guard in an acceptable format.

3 System model and methodology

In this section, we describe the security classification problem from the point of view of statistical natural language processing (SNLP), a text processing technique based on machine learning that is frequently used to perform topical classification. We propose a method of automatically assigning security classifications to unstructured texts (i.e., texts containing no explicit metadata such as author or date of creation), describing the system model and briefly reviewing all of the system components that must be optimized. Finally, we discuss the experimental technique and data sets we used to obtain our results, including a description of the documents used for the experiments.

3.1 The statistical natural language processing system model

The Government of Canada Security Policy [10] defines four levels of classification: Unclassified, Confidential, Secret, and Top Secret. The potential injury to the national interest arising from a document's unauthorized disclosure determines its level of classification. For the purposes of this paper, we consider only two security classification levels, $\mathcal{C} = \{\text{Unclassified}, \text{Classified}\}$, where Unclassified corresponds to the standard Security Policy definition and Classified corresponds to Confidential, Secret, and Top Secret documents. This limits us to the simplified case of a binary classification model.

In this work, we follow the standard SNLP approach to text categorization that treats each document as a bag-of-words, as discussed in [21]; under this model only the words and their relative frequencies are of interest in characterizing a document, as opposed to word order, parts of speech, and context. Further, this approach typically assumes that there exists a collection of documents—called a corpus—which is properly categorized and can thus serve as a training set for the supervised machine learning algorithms employed in the classifier. All the experiments discussed in this paper are based on the methodology depicted in Figure 1, which shows a block diagram of the classification system; each element is briefly described in the subsections below, with extensive descriptions of these techniques provided in a previous work [13].

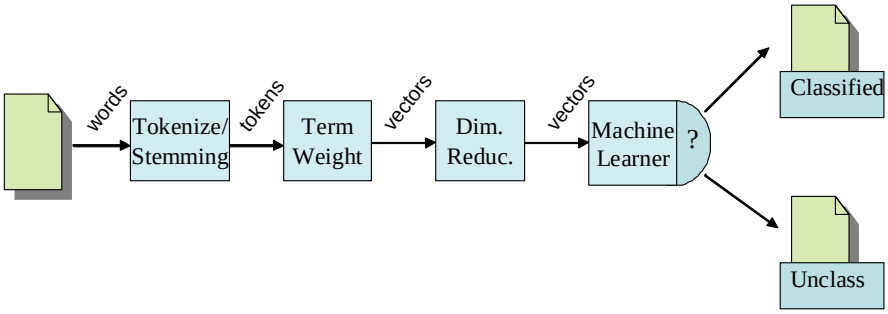


Figure 1: *Automated statistical security classification system model*

3.1.1 Tokenization and stemming

A document entering the system is decomposed into its constituent words, removing all stop-words in the process [3]. Next, a stemming algorithm such as the Porter stemmer, [22], is applied to convert all words to an appropriate root form (e.g., 'brought' and 'bringing' would be reduced to the root 'bring'). The stemmed words are then tokenized, meaning that the words are broken up into meaningful elements. Often tokens are simply the words themselves, but other possibilities exist; for instance, tokens could be N-grams (N words in conjunction) instead of single words in isolation.

3.1.2 Vectorizing the document

Once the document has been tokenized, it is converted into a vector. The j^{th} document in a collection, d_j, is represented by the vector $\mathbf{x_j} = \left(x_j^{(1)}, x_j^{(2)}, \ldots, x_j^{(n)}\right)$, where element $x_j^{(i)}$ is referred to as a token- or term-weight and represents the relative importance of token i in the document. Each document vector is n-dimensional, where n is the number of distinct tokens appearing in the entire corpus. The simplest term-weighting scheme is to use binary weights, where $x_j^{(i)} = 1$ if token t_i appears in d_j, and $x_j^{(i)} = 0$ otherwise. Another widely used term weighting scheme is the so-called *term frequency—inverse document frequency (tfidf)* scheme, which places a greater emphasis on tokens appearing multiple times in a document but less emphasis on terms that appear across many documents. The weights under *tfidf* are defined as

$$x_j^{(i)} = n(t_i, d_j) \cdot \log\left(\frac{|D|}{n(t_i)}\right), \tag{1}$$

3. Stop-words are common words—such as 'an', 'the', 'to', and 'for'—that are understood to have little predictive power in categorizing documents. It is assumed that these are removed before any further processing is done in order to reduce the processing burden.

where $n(t_i, d_j)$ represents the number of occurrences of token t_i in document d_j, $|D|$ represents the number of documents in the corpus, and $n(t_i)$ represents the number of documents in which token t_i occurs.

3.1.3 Dimensionality reduction

After tokenization, the document vectors are all of length n, where n is typically quite large since a corpus generally contains many thousands of tokens (i.e., words). Feature selection algorithms may be applied prior to classification in order to reduce the dimensionality of the vectors to a value $m < n$. This leads to a reduction in processing time for subsequent machine learning algorithms and in some cases may even *improve* categorization performance by removing "noisy" tokens that have no discriminatory or predictive value.

We consider two competing feature selection algorithms in this paper: chi-square (χ^2) and information gain. Extensive studies of feature selection for a variety of machine learners in [23] and [24] have identified these as the two most effective algorithms in achieving significant dimensionality reduction without sacrificing categorization accuracy. Both have the intended goal of identifying the tokens, or features, that are most predictive of a document's category; consequently, these dimensionality reduction algorithms are supervised to the extent that they rely on a labelled training set unlike other simpler techniques.

Chi-square feature selection computes the level of dependence for each token-category pair (t_i, c_j) by computing

$$\chi^2_{t_i,c_j} = \frac{|D| \cdot \left(n(t_i,c_j)n(\overline{t_i},\overline{c_j}) - n(\overline{t_i},c_j)n(t_i,\overline{c_j})\right)^2}{(n(t_i,c_j)+n(\overline{t_i},c_j)) \cdot (n(t_i,c_j)+n(t_i,\overline{c_j})) \cdot (n(\overline{t_i},\overline{c_j})+n(\overline{t_i},c_j)) \cdot (n(\overline{t_i},\overline{c_j})+n(t_i,\overline{c_j}))} \ , \tag{2}$$

where $n(t_i, c_j)$ represents the observed number of documents of category c_j in which word t_i appears and $n(\overline{t_i}, c_j)$ represents the observed number of documents of category c_j in which word t_i does not appear (and likewise for $\overline{c_j}$). The χ^2 statistic is marginalized over each category and the dimensionality is reduced to m by selecting the m tokens with the largest statistics.

The information gain for a token gives an indication of how many bits of information are gained towards predicting category c_j of a document by knowing that a token appears or does not appear in the document. The information gain for word t_i in category c_j is computed using

$$IG(t_i, c_j) = \frac{n(t_i, c_j)}{|D|} \cdot \log\left(\frac{|D| \cdot n(t_i, c_j)}{n(c_j) \cdot n(t_i)}\right) + \frac{n(\overline{t_i}, c_j)}{|D|} \cdot \log\left(\frac{|D| \cdot n(\overline{t_i}, c_j)}{n(c_j) \cdot n(\overline{t_i})}\right) \ , \tag{3}$$

with the notation as above for chi-square, and with the additional notation of $n(c_j)$ representing the number of documents in category c_j.

3.1.4 Supervised machine learning

A supervised machine learner develops a model based on a training set of labelled documents and uses this model to predict the classification of an unlabelled test document. By the time the documents reach the machine learner, all pre-processing—including tokenization, stemming, term weighting, and feature selection—has been completed and the documents are represented as m-dimensional vectors. We consider three popular machine learning algorithms, which have had significant success in the realm of topical classification: k-Nearest Neighbour (kNN), Naïve Bayes (NB), and Support Vector Machine (SVM) classifiers.

k-Nearest Neighbour

The k-Nearest Neighbour classifier has arguably the simplest algorithm. Based on a chosen "distance measure", the algorithm computes the distance between the test document vector and all training vectors. The k vectors closest to the test vector are selected and their labels are examined. The test document is assigned to the category shared by the majority of its k neighbours. In our experiments, we used the popular cosine similarity as a distance measure, computed as

$$\cos(d_i, d_j) = \frac{\sum_p x_i^{(p)} \cdot x_j^{(p)}}{\sqrt{\sum_p (x_i^{(p)})^2}\sqrt{\sum_p (x_i^{(p)})^2}}, \tag{4}$$

where $x_i^{(p)}$ denotes the weight of token p in document d_i.

Naïve Bayes

The Naïve Bayes classifier estimates the probability that document d_j belongs to class c_i by applying Bayes' rule to compute the probability

$$P(c_i|d_j) \propto P(c_i) \cdot P(d_j|c_i) = P(c_i) \cdot \prod_{1 \le k \le |d_j|} P(t_k|c_i), \tag{5}$$

where $P(c_i)$ is the *a priori* probability that any document belongs to category c_i; $P(d_j|c_i)$ is the probability that, given a document from category c_i, it is document d_j; and $P(t_k|c_i)$ is the probability that, given a document from category c_i, it contains token t_k. The naïve assumption of (5) is that the tokens in d_j occur independently, allowing us to write the conditional probability $P(d_j|c_i)$ as a product of token probabilities.

The above formulation of the Naïve Bayes classifier is known as the multinomial model; for our experiments we use the multinomial model with Laplacian smoothing, as discussed in more detail in [25].

Support Vector Machines

Although they can be extended to multi-category problems, Support Vector Machines are designed as two-category classifiers; either a test vector is a member of a group or it is not. Assuming m-dimensional document vectors, the training process of the SVM involves finding the $(m-1)$-dimensional hyperplane that maximally separates all elements from the two categories, i.e., it is the hyperplane that provides the widest margin between itself and any training vector on either side.

A hyperplane is defined by the equation $\mathbf{w} \cdot \mathbf{x_j} + b = 0$, where $\mathbf{w}$ is the normal vector to the hyperplane and $b/\|\mathbf{w}\|$ is its distance from the origin. To determine the classification of a document vector $\mathbf{x_j}$, the SVM evaluates $y_j = \text{sign}(\mathbf{w} \cdot \mathbf{x_j} + b)$; the sign of the result, y_j, indicates on which side of the hyperplane the document vector is located and thus indicates its predicted category.

Training the SVM requires the computation of $\mathbf{w}$ and b from the set of training vectors such that the margin is maximized, which can be expressed as the following optimization:

$$\begin{aligned} \min_{w,b} \quad & \tfrac{1}{2}\|w\|^2 \\ s.t. \quad & y_j \cdot (\mathbf{x_j} \cdot \mathbf{w} + b) \geq 1. \end{aligned} \tag{6}$$

Further details on the SVM, including Quadratic Programming methods to solve the optimization in (6) can be found in [26, 27].

3.1.5 A classifier's confidence level

Typically, the output of the *binary* classifiers discussed in Section 3.1.4 is a decision assigning a document to one of two categories. A binary decision is black or white—the document is either in one category or the other. Not normally mentioned is the fact that each of these learners can assign a "level of confidence" to the decisions it makes about a document's categorization. This fact plays an important role in our analysis in Section 4.6 and bears some discussion here.

The term "confidence" in the context of a classifier's confidence level is unrelated to the idea of statistical confidence intervals. It is entirely a by-product of the algorithm the classifier uses to render its decision. A k-Nearest Neighbour classifier, for instance, is more confident in its decision if *all* k neighbours of the test document have the same classification, as opposed to a case where the neighbours are split evenly among the possible categories.

Methods for computing each classifier's level of confidence are summarized below:

- **k-Nearest Neighbour:** Confidence measure that document d belongs to category c is calculated as the percentage of the k neighbours from the training set that belong to category c. This is essentially reporting the results of a survey of the documents "most similar" to the test document under consideration.

- **Naïve Bayes:** Confidence measure that document d belongs to category c is $\frac{P(c)P(d|c)}{P(\overline{c})P(d|\overline{c})+P(c)P(d|c)}$. Assuming that the Naïve Bayes assumption is correct [4], this equation computes the probability that the document in question is classified. The computations are normalized such that the probabilities of a document being classified or unclassified will indeed add to one.

- **Support Vector Machine:** Confidence measure that document d belongs to category c is the distance of the document vector d from the hyperplane that maximally separates the training set. Note that the distances are normalized such that the maximum possible distance of a document from the hyperplane is one. Intuitively, a document that is further from the hyperplane will be more likely to be classified correctly.

3.2 Experimental methodology

In order to obtain statistically significant results when evaluating the performance of any text categorization system, a large labelled corpus of documents is required [5]. This section describes the corpora of documents that we used throughout our experiments and discusses n-fold cross-validation, a standard technique used to evaluate the accuracy of a text categorization system.

3.2.1 Test data sets

Carrying out experiments on a large corpus of classified government documents is impractical due to the obvious difficulty in procuring a sufficient number of documents and the restrictive handling procedures such documents require, not to mention that any results would themselves be classified and could not be openly published. For our experiments, we have obtained several hundred declassified government documents from the Digital National Security Archive (DNSA) [4]. This archive is a valuable resource, housing electronic copies of over 80,000 previously classified (and unclassified) government documents relating to U.S. foreign policy from the post-World War II era to present day. Originally intended as a resource for foreign policy research, it is well suited to our research as well.

4. Recall that the so-called naïve assumption is that all words in the document are mutually independent.

5. It is not unusual for such corpora have thousands of documents.

Many documents from the DNSA retain their original historical labels, so although they are all now currently declassified, their original classifications are known. This is ideal for the experiments we conduct, since we can obtain a labelled set of documents of varying degrees of sensitivity (from Unclassified to Top Secret). In this paper, we assume the documents are correctly labelled. This assumption is necessary to carry out our work, but we should point out that it is likely some of the documents would indeed be mislabelled due to flaws in the classification process when these documents were created. Ultimately, the fidelity of a decision made by a machine learner is tied to the reliability of the data used to train the learner.

The DNSA sorts documents according to "collection", whereby documents related to U.S. foreign policy towards a given nation during a given era would be in its own collection. Initially, we retain this division, examining the ability to discern security classification *within a collection*.

For our initial experiments we focus primarily on three specific collections: U.S. Foreign Policy towards the Philippines (1973-1981), U.S. Foreign Policy towards Afghanistan (1980-1988), and U.S. Foreign Policy towards China (1990-1999). These collections were selected specifically because they contained a relatively large number of documents that were nearly evenly split between Unclassified and Classified (i.e., Confidential, Secret, and Top Secret). In no collection were there *exactly* the same number of Unclassified as Classified documents, so where necessary we randomly selected a small number of documents to exclude from our experiments to ensure that the number of documents from each category were equal and that there was no bias induced by an uneven training distribution [6]. We used the abstracts of the documents to conduct our experiments. This is consistent with the standard practice in text categorization literature of using short documents or abstracts such as the Reuters-21578 corpus (newswire stories) or the Oregon Health Sciences University Medicine (OHSUMED) corpus of medical abstracts, [28–30]. Table 1 details the statistics regarding the three corpora we used, indicating the number of documents, and the number that were Classified and Unclassified.

Table 1: *Corpus Statistics*

Corpus	# Docs	# Unclass	# Class
Afghanistan (1980 - 1988)	248	124	124
China (1990 - 1999)	176	88	88
Philippines (1973 - 1981)	262	131	131

It is important to note that the documents we consider in our experiments are all

6. For example, in the Philippines corpus we initially had 137 Unclassified documents and 131 Classified documents. We randomly selected 6 Unclassified documents for removal from the corpus.

foreign policy documents. Certainly, not all documents we would be faced with in a real-world scenario would conform to this style. In fact, some documents could prove easier to classify—for instance technical documents where a clear rule is established such as "all IP addresses are classified". We believe, however, that the primary value of SNLP methods is precisely for those cases where clear rules do not apply and inference must be used to determine the sensitivity of a section of text.

3.2.2 Training and testing using n-fold cross-validation

For all our experiments, we used stratified n-fold cross-validation to evaluate the performance accuracy of each classifier; this technique is standard practice in text categorization literature (see, for instance, [21, 24, 31]). With this technique, the corpus is divided into n sets, where each set contains an equal number of documents [7]. Documents are randomly assigned to each set with the only restriction being that the sets are constructed such that the distribution of the documents in the set parallels the distribution of the corpus (meaning in our case that since each of our corpora are half Unclassified and half Classified, half of the documents in each of the n sets will be Classified as well). Finally, a single set is retained as a test set and the remaining $n-1$ sets are used to train a classifier; the trained classifier is evaluated using the retained test set. This is repeated n times such that each of the n sets are used as a test set exactly once. The accuracy estimate is given by the average of the n test runs.

3.2.3 Adding confidence intervals

To add confidence intervals to our results and to select an appropriate value for n in our cross-validation, we follow the methods suggested by Kohavi in [32] based on his extensive experimental investigation of accuracy estimation for supervised classification algorithms [8]. Kohavi recommends that to obtain a low-bias, low-variance accuracy estimate, the best approach is to use stratified n-fold cross-validation with $n = 10$.

To compute the confidence intervals of a single data point, we use the percentile method of [33], recommended in [32], wherein we run 10-fold cross-validation 25 times, with a different randomization mapping documents to the n sets in each run. The accuracy is reported as the median sample (the 13th largest sample), with high and low estimates given as $+/-1$ standard deviation (the 5th and 21st largest samples respectively).

7. In most cases the number of documents in the corpus will not be evenly divided by n. In this case each set will contain the same number of documents as any other +/-1 document.

8. Kohavi's discussion of accuracy estimation on pp. 35-75 of [32] is very thorough and provides valuable insights into the challenges of estimating the accuracy of a classifier.

4 Experiments and discussion of results

This section presents and analyzes the results of a series of experiments we conducted, investigating the application of the system depicted in Figure 1 to the task of assigning security classifications to unlabelled documents. The experiments are intended to study the effects of changes to each of the system blocks and to learn how changes to a training set will affect our model's accuracy. We present the following eight distinct experiments to investigate our system:

- **Experiment 1—Baseline:** This experiment evaluates the performance of the system in Figure 1 under very simple circumstances to establish a baseline performance metric.

- **Experiment 2—Term weighting and dimensionality reduction:** The intent of this experiment is to quantify the effect of dimensionality reduction on classification accuracy and to determine which term weighting scheme (*tfidf* or binary) produces better results.

- **Experiment 3—Tokenization:** Results from this experiment are used to optimize the value of N when performing N-Gram tokenization.

- **Experiment 4—Stemming:** This experiment investigates whether or not the use of stemming improves classification accuracy.

- **Experiment 5—Choosing a machine learner:** The performance accuracy of the machine learners is directly compared and a framework for selecting the best machine learner is developed.

- **Experiment 6—Classifying documents as "unknown":** In this experiment, the classifier is permitted to leave a document unlabelled if the document proves difficult to categorize. We examine the effect of this condition in detail.

- **Experiment 7—Cross-domain learning:** This experiment investigates the accuracy degradation experienced by the system when documents from a new topical domain are introduced (i.e., the topic domain of the test documents differs from the topic domain of the training documents).

- **Experiment 8—The temporal effect:** The intent of this experiment is to investigate the impact of temporal correlation between training data and test data.

Note that for all experiments, the open source data mining software package RapidMiner version 4.5 and its associated Text Processing Plug-in were used to implement

the machine learners and requisite preprocessing.

4.1 Experiment 1: Baseline

To establish a baseline, we conducted an experiment to estimate the accuracy of each of the three machine learning classifiers from Section 3.1.4 (k-Nearest Neighbour, Naïve Bayes, and SVM) on each of the corpora (Afghanistan, China, and Philippines) described in Section 3.2.1. A classifier's accuracy is defined as the percentage of decisions it makes that are correct. For our 2-category problem, this is

$$\frac{np(c_1, c_1) + np(c_2, c_2)}{np(c_1, c_1) + np(c_2, c_2) + np(c_1, c_2) + np(c_2, c_1)}, \tag{7}$$

where $np(c_i, c_j)$ is the number of test documents from category c_i that are predicted to be in category c_j.

For this experiment, there was no stemming algorithm applied to the words, a *tfidf* term weighting was used for the tokens, and dimensionality was not reduced. Figure 2 shows the results of this experiment [9].

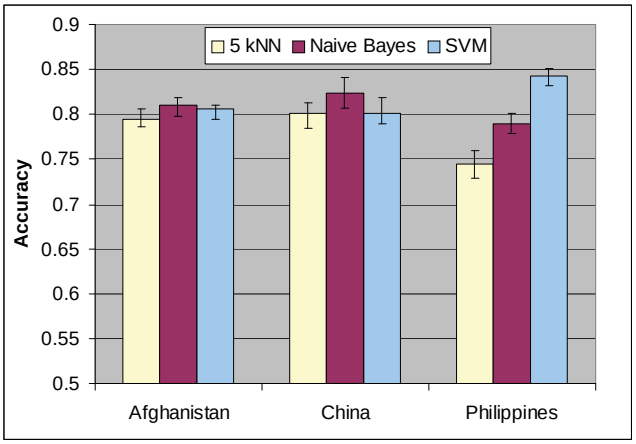


Figure 2: *Classification accuracy:* tfidf *term weights, no stemming, no dimensionality reduction*

As discussed in Sections 3.2.2 and 3.2.3, 10-fold cross validation was used to train and test the learners and the confidence intervals shown represent the +/- 1 standard deviation, computed using the percentile method as in [33]. All subsequent experiments utilize the same technique.

9. Note that the k-Nearest Neighbour classifier used in our experiments has k = 5. This value for k was chosen based on several initial experiments on the Afghanistan, China, and Philippines corpora. The results of these initial experiments are reported in Annex A. Within experimental error, no performance improvement (i.e., classification accuracy improvement) was noted when k was increased beyond 5. In fact, in some cases performance began to degrade.

The results of our baseline experiment shown in Figure 2 are encouraging: in all cases, the machine learning models are able to correctly identify a document as "classified" versus "unclassified" at least 70% of the time; in 8 out of 9 cases the rate is better than 75%. We defer further analysis and discussion of the performance and merits of the various machine learners on each corpus to Section 4.5, where further results are presented along with a framework for evaluating the classifiers.

Note that equation (7) focuses on the *total* classification accuracy of the schemes—this is the format pursued throughout our experiments and is in line with typical literature in the field of binary categorization (accuracy is the preferred metric for binary problems, e.g., [31, 34]). Total accuracy encompasses classified documents that are miscategorized as unclassified as well as unclassified documents erroneously categorized as classified. Table 2 shows a separation of these error modes for the baseline case, with the "Class Accuracy" column indicating the accuracy rate for classified documents and the "Unclass Accuracy" column indicating the accuracy rate for unclassified documents. The average accuracy is the average of these two columns.

Table 2: *Overclassification and Underclassification in baseline results*

Corpus	Learner	Average Accuracy	Class Accuracy	Unclass Accuracy
Afghanistan	kNN	79.4%	83.1%	76.6%
Afghanistan	NB	81.0%	87.1%	75.0%
Afghanistan	SVM	80.6%	88.8%	72.6%
China	kNN	80.1%	80.7%	80.1%
China	NB	82.4%	83.0%	82.4%
China	SVM	80.1%	83.0%	77.3%
Philippines	kNN	74.4%	85.5%	64.1%
Philippines	NB	79.0%	93.1%	64.9%
Philippines	SVM	79.0%	83.2%	85.5%

Of note is that it is possible to bias the classifiers in one direction or the other, meaning that we could increase the accuracy rate for classified documents at the expense of a decreased accuracy rate on unclassified documents. For the remainder of our work, however, we do not bias the classifiers in one direction or the other, focusing instead on maximizing total classification accuracy. System bias is a tunable parameter that could be adjusted once a satisfactory classifier is selected.

4.2 Experiment 2: Term weighting and dimensionality reduction

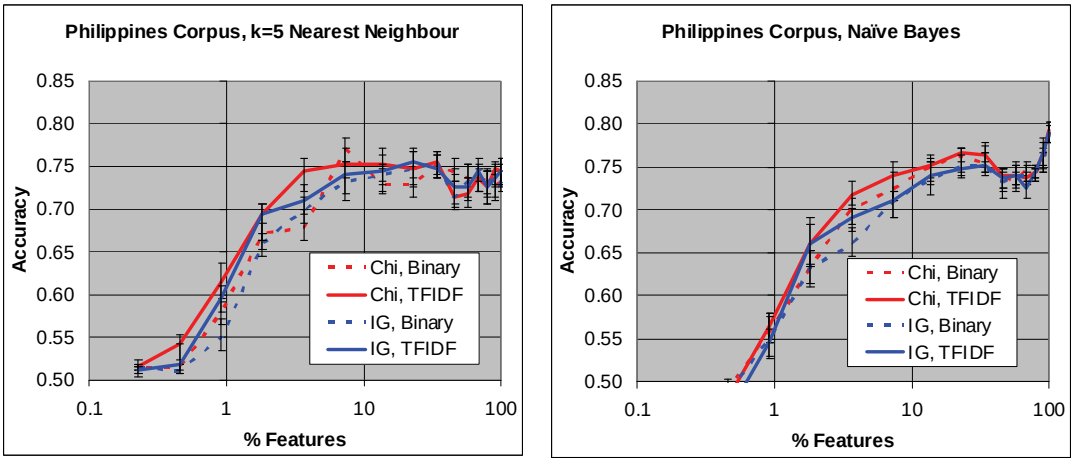
These experiments were intended to investigate the impact of term weights on classification accuracy and to determine the performance penalty incurred by reducing the document vector sizes. For term weighting, we compared simple binary weights with the more intuitive (but more complex) *tfidf* scheme; Pang and Li [34] reported the somewhat surprising result that *tfidf* offered no benefit for sentiment classification and we wished to determine if the same holds true for security classification. We examined both the chi-square and information gain feature selection techniques for dimensionality reduction, as these are consistently reported to be the most robust techniques from the point of view of allowing a classifier to retain accuracy while aggressively reducing the feature space.

4.2.1 Dimensionality Reduction

Figure 3 shows the classification accuracy as a function of the percentage of features retained; thus, if the dimensionality of the document vectors is reduced by $x\%$, then the percentage of features retained would be $(100 - x)\%$. Note that while Figure 3 provides the results of experiments on only the Philippines corpus, similar results were obtained for the Afghanistan and China corpora with these results provided in Annex A for reference; the following discussion applies equally well to those results. The effects of term weighting and dimensionality reduction are considered for three separate machine learners, with Figure 3(a) showing k-Nearest Neighbour (with k = 5), Figure 3(b) showing Naïve Bayes, and Figure 3(c) showing a Support Vector Machine classifier.

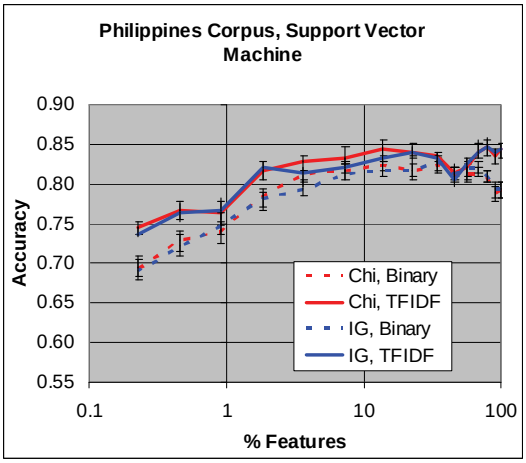
For all three classifiers we observe a general downward trend in the classification accuracy as the percentage of features retained is reduced. However, the degradation in performance is very modest—we observe a loss of less than 5% accuracy even when as few as 10% of the features remain. As the dimensionality is reduced below the 10% mark, the performance of all three classifiers tends to degrade more rapidly with the most marked degradation exhibited by the Nearest Neighbour and Naïve Bayes classifiers. Furthermore, we observe no major performance difference between the information gain technique (blue line in Figure 3) and chi-square technique (red lines in Figure 3) throughout the range of retained features. Chi-square does appear to have a slight edge over information gain, but this is within the +/- 1 standard deviation confidence intervals for all but two points.

These results are largely consistent with observations in [23] and [24], where Yang *et. al.* examined dimensionality reduction techniques for topical classification; the chief difference between our results and Yang's is that Yang reported an inferior



(a)

(b)



(c)

Figure 3: *Classification accuracy of documents from the Philippines corpus*

performance of chi-square (compared to information gain) under extremely aggressive feature selection (less than 1% of features retained), whereas we observe no such difference in performance. In any case, it is unlikely that one would wish to design a classification system that reduces the dimensionality by over 99% as this would result in significantly impaired performance regardless of the feature selection technique used.

4.2.2 Term Weighting

The effect of term weighting is also shown in Figure 3, where the dotted lines represent results obtained using binary weights and the solid lines represent results obtained using *tfidf* weights. While [34] reports no benefit to using *tfidf* for sentiment classifica-

tion, results in [35] strongly support using *tfidf* over any other term weighting scheme for topical classification when the documents to be classified are relatively short (as is the case with our corpora). Yu, [31], reports that ultimately *tfidf* is strongly task-dependent and the only way to truly determine whether or not it is suitable for a classification problem is to run the experiment.

For the security classification exercise, we observe that in general when the percentage of retained features is high there appears to be little distinction between the *tfidf* weighting scheme and simple binary term weights. Security classification appears to share this in common with sentiment classification, which stands in contrast to topical classification where the repetition of key words is a strong indicator of the topic of a document. The simple presence or absence of a word provides sufficient information for security classification when all (or most) features are retained. Interestingly, however, a noticeable pattern across all corpora suggests that *tfidf* becomes slightly more beneficial as feature selection becomes more aggressive. A general rule of thumb, then, is that *tfidf* increases classification accuracy for low dimensionality and is benign for higher dimensionality.

One possible hypothesis for why the term weighting is less important when all features are retained is that *related and co-located* words provide equivalent useful information similar in value to the *tfidf* weight. As an illustrative example, in the Afghanistan corpus one of the words with the highest discriminatory power is "refugee" (in fact, this word has the highest discriminatory power in the corpus). Often other words co-occur with "refugee" such as the words "camp", "repatriation", and "relief"; however none of these co-located words is among the top 20 discriminatory words. If *all* the features in the corpus are retained, then the trained models will take all these words into account: a model generated in this fashion may incorporate the fact that the occurrence of the words "camp", "repatriation", and "relief" already give a strong sense that the term "refugee" plays a major role in the document. In this case, the additional strong *tfidf* weight on "refugee" is redundant. However, if we prune the features such that we select only the top 20 discriminatory terms, then knowing whether or not "refugee" has a strong weight in a particular document is *very* relevant since it alone (without the help of co-located terms) indicates the importance of the word in that document.

4.3 Experiment 3: Tokenization

The bag-of-words model does not take into account word order, parts of speech, or sentence structure, instead treating every word as a token (or element) in the vector model of a document. The so-called N-Gram tokenization technique can be introduced to this model to place *some* emphasis on word order. An N-Gram is a token that represents a collection of N words from the document in the order that

they appear. So, for instance, if we wanted to perform 2-Gram tokenization (or bigram tokenization as it is more frequently called) on the phrase, "give Homer a donut", the resulting tokens would be as follows: "give Homer", "Homer a", and "a donut".

We investigated the effect of N-Gram tokenization on classification accuracy for each of the three machine learners, up to $N = 5$. We used *tfidf* term weighting, performed no dimensionality reduction, and performed no stemming. The results of this experiment are shown in Figure 4.

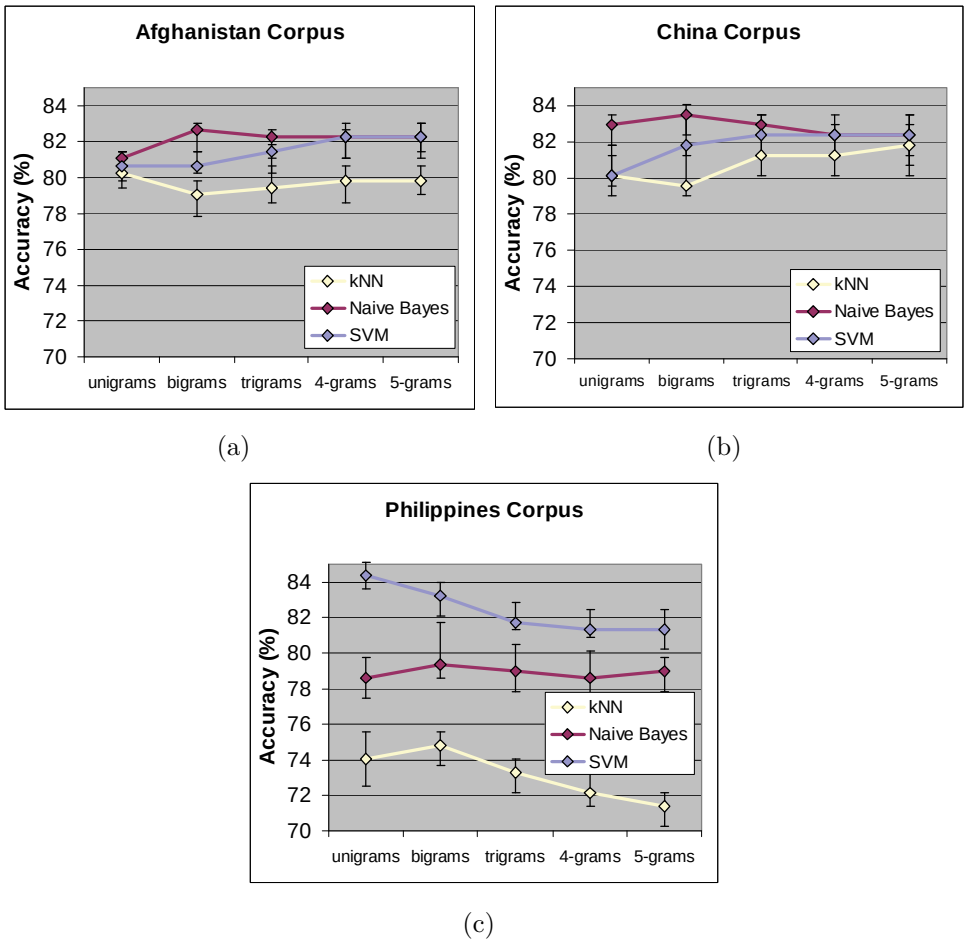


Figure 4: *Performance of the machine learners after N-Gram tokenization.*

Our results indicate that within the +/- 1 standard deviation confidence interval, N-Gram tokenization does not appear to offer significant improvements in performance over using simple unigram tokens. A notable exception is in the China corpus where the kNN classifier and the SVM classifier obtain marginal benefits for 3-, 4-, and 5-Gram tokenization; this trend does not extend to the other corpora, however. These results are consistent with observations in standard topical classification (first

reported in [36], where meaningful bigrams provided no benefit to a topical classifier) and sentiment classification (e.g., [34]). Interestingly, the additional context provided by word order does not seem to have a significant impact on the classification task; the unigrams alone form a sufficient statistic for rendering a classification decision.

4.4 Experiment 4: Stemming

Our next experiment was intended to investigate the value of stemming—that is, to determine whether or not our accuracy improves by reducing words with a common root to a single token. To investigate the effect of stemming on classification accuracy, we implemented the popular Porter stemming algorithm [22] and compared the accuracy of all three learners *with* stemming and *without*. Once again, we used a *tfidf* term weighting and performed no dimensionality reduction.

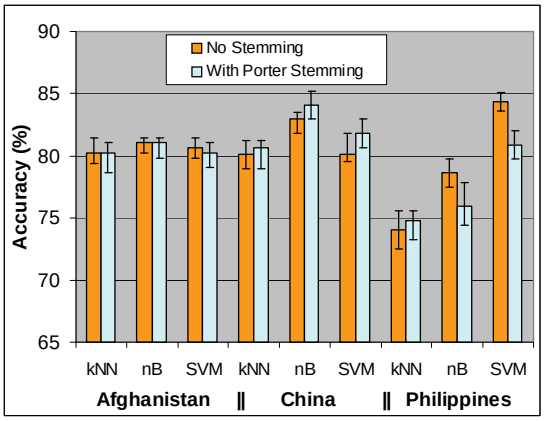


Figure 5: *Effect of stemming on classification accuracy.*

In 5 out of 9 cases, the use of stemming actually results in poorer performance than when stemming is not used—although in one of these cases (SVM on the Philippines corpus) the performance difference is larger than +/- 1 standard deviation. In the remaining 4 out of 9 cases, stemming offers an improvement, but in all cases this is within the margin of error. Thus, based on our experiments, stemming is of little value and is not recommended.

4.5 Experiment 5: Choosing a machine learner

While the most obvious way to select among machine learning algorithms is to compare their relative classification accuracies, this should not be the only consideration. We propose the following five criteria to evaluate the effectiveness of a machine learner for the task of assigning a security classification to unlabelled text:

1. **Accuracy:** Which learner has the greatest classification accuracy on each corpus?
2. **Consistency:** We prefer a learner that is reliably accurate as opposed to a learner that is very accurate on some corpora and not on others. Are some learners more consistent than others?
3. **Adaptability:** A machine learner develops a model based on training data. What is the processing time required to develop the model? As more data become available, how quickly or easily can the model be adapted to reflect the new data?
4. **Complexity:** Which learners require the most / least processing power to classify a new document?
5. **Transparency:** How easily explained are the classification decisions made by a machine learner?

We evaluated the accuracy and consistency of the machine learners through empirical experiment, while the adaptability, complexity, and transparency were evaluated by considering in detail how each learner operates. This section begins with a discussion of our empirical experiments comparing the learners (adaptability and consistency), followed by a non-empirical discussion of adaptability, complexity, and transparency. The section concludes with a summary of all criteria, presented in Table 3, along with some general recommendations about choosing a machine learner.

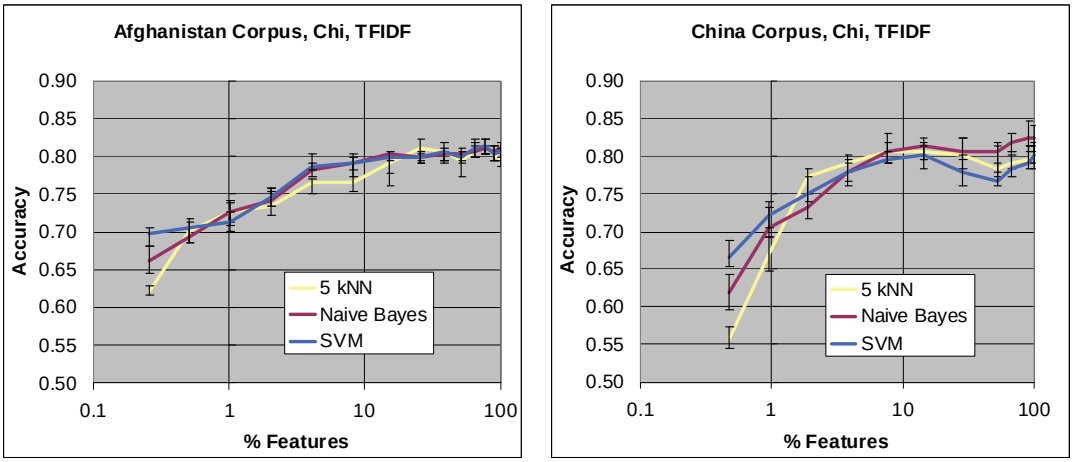
4.5.1 Empirical results: accuracy and consistency

Our experimental results focus on determining the accuracy and consistency of the machine learners we considered. These are detailed below and summarized in Table 3 in Section 4.5.3.

Machine learning accuracy

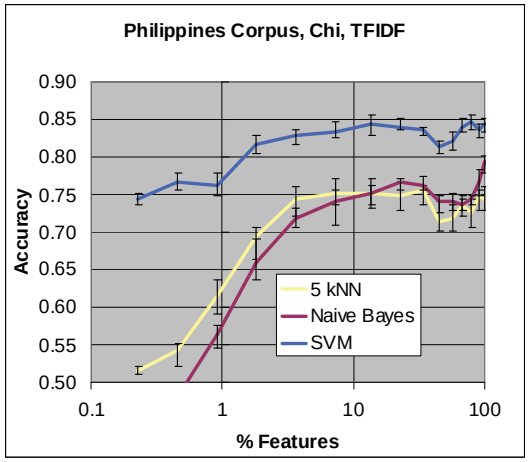
The most obvious metric to use in selecting among machine learning algorithms is to compare their relative classification accuracies. Since we are interested in how the learners perform in conjunction with judiciously chosen term weights and feature selection, we applied *tfidf* weighting and a chi-square feature selection algorithm based on the results in Section 4.2. We performed unigram tokenization and did not employ stemming, as per the results in Sections 4.3 and 4.4. Figure 6 plots the classification accuracy of each learner as a function of the percentage of features retained.

Figure 6(a) shows that all three learners have comparable performance on the Afghanistan corpus in the high-dimensionality case, obtaining better than 81% accuracy. As the percentage of features retained is reduced below 10%, both SVM and Naïve Bayes



(a)

(b)



(c)

Figure 6: *Comparison of learners assuming chi-square feature selection and* tfidf *weighting.*

have a performance edge over kNN. SVM is known to be relatively insensitive to feature selection and this is clear here. In the China corpus, depicted in Figure 6(b), Naïve Bayes is the most accurate learner for high dimensionality (with an accuracy of 82%), with SVM taking the lead as dimensionality reduction becomes more aggressive, once again demonstrating the robustness of SVM to feature selection. For the Philippines corpus, SVM is the clear winner over kNN and Naïve Bayes as shown in Figure 6(c), SVM having an accuracy of 84% over Naïve Bayes at 80%. A notable conclusion observed in all three corpora is that the Nearest Neighbour learner is never the best choice for any corpus at any level of dimensionality reduction [10].

10. A single data point in the China corpus has kNN as the performance leader, but there is no general trend here.

The general wisdom from topical classification is that SVM typically outperforms all other learners (in fact, it has been difficult to surpass the accuracy of SVM for topical classification since its first introduction to the problem in [28]). Research in non-topical classification, however, has been less definitive, with [31] making a strong case for Naïve Bayes in identifying the predominant emotion in works of literature.

In our experiments, while the SVM learner does show dominance on the Philippines corpus, the results from the other two corpora do not suggest that SVM is a clear-cut choice. For security classification, our research suggests that SVM and Naïve Bayes have comparable classification accuracy, with kNN running a distant third. Interestingly, neither SVM nor Naïve Bayes is a clear winner on every corpus making it difficult to know which learner to rely on. In Section 4.6 we propose a method to combine the results of an SVM and Naïve Bayes classifier in order to get the best of both worlds.

Consistency of learners

Apart from the relative accuracy of the machine learners, their consistency is also of interest. A learner that performs reliably well is of more practical value than a learner that performs very well on some data and poorly on others. Figure 7 depicts the effect of the choice of corpus on the performance of each learner.

To comment on the consistency of a learner, we examined the accuracy spread of the learners. We computed the accuracy spread as the difference between the maximum and minimum accuracies of a learner among the three corpora; for instance, the accuracy spread of SVM with 100% of features retained is the difference in accuracies of the Philippines corpus (maximum accuracy) and China corpus (minimum accuracy). The spread is plotted in Figure 7(d).

We observe that the SVM has the smallest spread across the entire range of dimensionality reduction. The kNN and Naïve Bayes learners are relatively close under moderate dimensionality reduction (more than 10% of features retained) but begin to diverge as dimensionality reduction becomes more aggressive with Naïve Bayes exhibiting a significantly larger spread. Although Naïve Bayes has the largest spread, it is not particularly problematic since the large spread is primarily apparent in the lower dimensionality ranges beyond where the system is likely to be operated.

The consistency of a learner's performance is not a criteria that is typically studied in the literature. While [37], for instance, investigates a sentiment classification problem across multiple domains, there is no discussion of the consistency of the results across corpora. When applied to security classification, however, it is important to know whether the chosen learner can be relied upon with some degree of confidence. To know simply that a learner is better than the competition is not sufficient if all the

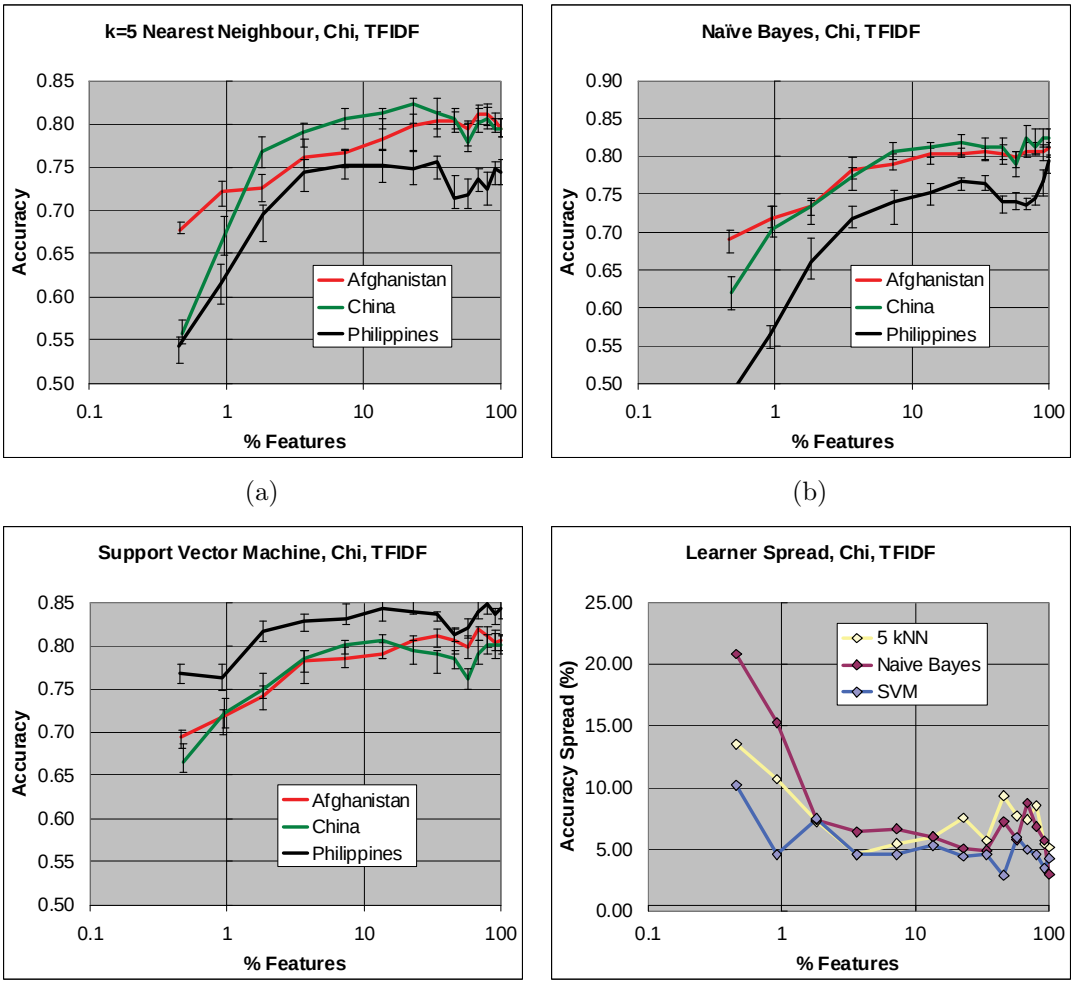


Figure 7: *Accuracy for each individual learner shown in (a), (b), and (c); the spread of all learners shown in (d).*

learners perform universally poorly on a certain corpus. While our study of three corpora has provided initial data suggesting the greater consistency of SVM over kNN and Naïve Bayes, to develop a more rigorous determination of the consistency of the learners would require significantly more corpora and data.

4.5.2 Non-empirical results: adaptability, complexity, and transparency

In addition to the empirical results of Section 4.5.1, we consider additional criteria that can be evaluated without the need for experiment: adaptability, complexity, and transparency. These are detailed below and summarized in Table 3 in Section 4.5.3.

Adaptability of learners to new data

The training time of a machine learning algorithm is important since it tells us how easy it is to update a model as new data becomes available. If our model was static and there was never any need to change it once it had been trained, training time would be a negligible consideration. The real world is not static, however, as documents become declassified on their declassification date or when the data they contain is no longer relevant. Furthermore, as security policy evolves, so too must the models. In the extreme case, a model must be able to be retrained "from scratch" if a significantly large component of the training data has been "overwhelmed by events". We see the value in an intelligent model that can adapt to new information provided by users; a security classification suggested by an automated system could be accepted or rejected by a knowledgable user and this feedback could be used to re-train the learner.

The simplest learner for training purposes is kNN; in fact kNN has no training phase at all since it does not pre-compute a model; it classifies a new test instance by comparing it to each element in the training set. Thus its training complexity is $O(1)$, making it very amenable to new training instances.

The Naïve Bayes learner has training complexity $O(|D_{train}|\cdot t_{avg}+|C|\cdot n)$, where t_{avg} is the average number of terms in a document, $|D_{train}|$ is the number of documents in the training set, $|C|$ is the number of categories [11], and n is the document vector length [25]. This derives from the fact that a single pass through all the terms in the corpus is required to extract and count the terms, with average complexity $O(|D_{train}|\cdot t_{avg})$, followed by the cost of computing the $|C|\cdot n$ conditional probabilities. Adding a new document to the training set would not entail counting over all existing documents, making the complexity of updating the learner $O(|C|\cdot n)$. While this is obviously worse than kNN (no training time), it is linear with the document vector length, making it very manageable.

Standard estimates for SVM training time give a complexity of $O(|D_{train}|^3)$, where $|D_{train}|$ is the number of documents in the training set [38]. Researchers have had some success in reducing this time complexity in certain instances, although the time is still non-linear in $|D_{train}|$; optimal SVM training algorithms are still an open problem and an active area of research. The computational complexity for training an SVM is by far the largest when compared to Naïve Bayes and kNN.

Machine learning computational complexity

While the previous discussion focused on the time complexity of training a machine learner, we are also interested in the time complexity required to classify a new test document. This is arguably a more important criteria than the training complexity,

11. In our case we have only two categories: classified and unclassified.

since the classification task will most certainly be repeated more frequently than the training task in any conceivable implementation.

To classify a new document, both SVM and Naïve Bayes run in linear time relative to the number of tokens, with SVM having complexity $O(n)$ and Naïve Bayes having complexity $O(|C| \cdot n)$ (see [38] and [25]). While kNN required no training time at all, it pays for this by having the highest classification complexity at $O(|D_{train}| \cdot n)$, explained by the fact that it must evaluate the cosine similarity between the test document and each of the training documents. This is a significant disadvantage of kNN since the classification time grows progressively worse as the training set grows, which will inevitably be the case.

Transparency of the decision method

In our experiments, the output of a machine learner is a binary decision: a document is either classified or it is unclassified. However, for any given test document, each learner takes very different steps to arrive at this decision. When we refer to the "transparency" of a machine learner, we are referring to how intuitively satisfying (and how simple) the explanation is that describes the machine learner's decision. Arguably, this is an important consideration if a machine learner is incorporated into a system that is used as a manual aid in assigning security classifications. If a machine learner assigns a classification that differs from the original classification or from the opinion of a user, there must be human understandable feedback for the security officer or user to review.

As discussed in Section 3.1.4, the Naïve Bayes algorithm estimates the conditional probabilities $p(t_k|\text{classified})$ and $p(t_k|\text{unclassified})$ [12] for all tokens or words appearing in a document as a means of computing the probability that a document should be classified. The decision of the algorithm could easily be supported with a simple statement to a user pointing out the presence of several terms that indicate that the document is sensitive (or not). This relative simplicity appeals to our intuition; indeed, in a study of machine learning as an aid for diagnostic medicine, Kononenko reported in [39] that diagnostic tools based on a Naïve Bayes classifier were simple for physicians to understand and mimicked the physicians' method of arriving at a diagnosis.

Likewise, the kNN learner offers a simple explanation of a classification decision. While the learner is actually basing its decision on the cosine similarity between the vector representations of the test and training documents, this could be presented to a user as the idea that the test document in question is "most similar to the following

12. Where $p(t_k|\text{classified})$ denotes the probability that a document contains token t_k given that it is classified.

k documents." Since the similar documents are all classified (or unclassified), it makes sense that the test document adopts the same label.

Unlike the Naïve Bayes and kNN learners, the reasons behind an SVM decision are less transparent. A user would be less likely to appreciate an explanation discussing the relation of the vector representation of a test document to the hyperplane that maximally separates support vectors from the two classes. In his review of classifiers, Kotsiantis [40] reports that SVM has "notoriously poor interpretability."

4.5.3 Summary of machine learning evaluation

Table 3 provides a summary of the relative performance of the three machine learners based on the criteria discussed in the previous sub-sections. The importance of these criteria as they relate to the problem of security classification was discussed above. It is important to note that the criteria are not of equal weight; depending upon system constraints and requirements, certain criteria may be deemed more important than others. The table uses the following notation to evaluate the learners: ● = Good, ◐ = Fair, ○ = Poor. The rankings in the table are meant to convey relative performance, as opposed to absolute performance; for instance, it could be that the accuracy of all three learners is deemed inadequate but the one with the highest accuracy will receive a rank of "Good" in the table.

Table 3: *Comparison of Machine Learners*

	k-NN	Naïve Bayes	SVM
Average accuracy	◐	●	●
Consistency of results	◐	○	●
Adaptability to new data	●	◐	○
Classification time / computational complexity	○	●	●
Transparency (ease of interpretation)	◐	●	○

A quick glance at Table 3 reveals that there is no silver bullet when choosing a machine learner. Depending upon specific system goals, some learners are more suitable than others. However, for an application that is relied upon to give advice about the security classification of a document, Naïve Bayes has the most attributes to be recommended, namely: it is comparable in accuracy to SVM, has a reasonable computational complexity, is amenable to accommodating new training data, and can be relied upon to deliver a reasonably intuitive rationale for its decisions.

4.6 Experiment 6: Classifying documents as "unknown"

Section 4.5.1 focused on the classification accuracy of three machine learning algorithms, where the classifiers assigned a label of "classified" or "unclassified" to a document. We conducted another experiment to investigate the possibility of allowing the classifier to return a decision of "unknown" if it was unable to determine a document's classification with a significant level of certainty.

Recall from Section 3.1.5 that as a by-product of the machine learning algorithms, each classifier can generate a metric indicating its level of confidence on any decision. Our hypothesis was that if we allowed a classifier the flexibility of returning a decision of "unknown" when it was uncertain about a result, we would observe an increase in the accuracy of the classifier's output since it would make decisions only on documents for which it had a high level of confidence.

Figure 8 shows the error rate of each classification scheme when the least reliable documents are left unlabelled—that is, when the documents with the lowest confidence measure are classified as "unknown". The error rate of each learner is plotted as a function of the percentage of documents classified as "unknown". For instance, when 0% of documents are left with an unknown label, then all documents are assigned a classification and the error rate is at its peak. As the percentage of unlabelled documents increases (where the unlabelled documents are those for which the classifiers have the least confidence), the error rate decreases, as we would expect.

Of most interest is the fact that the error rate of *labelled* documents can be decreased significantly by opting to leave selected documents unlabelled. It is possible to reduce the error rate of certain classifiers well below 10% while still assigning a label to over two thirds of the documents. This is an important result; a classification system that is expected to provide guidance on a document's level of sensitivity would be valuable only if its decisions could be trusted. A system capable of indicating that it does not have a sufficient level of confidence to make a decision is of much more value than a system that consistently makes a large number of errors.

While Figure 8 provides a visual indicator of the benefits of incorporating the learner confidence measures, a simple metric to evaluate each learner individually is to note the percentage of documents left unlabelled at several key error rates. We are interested in the percentage of documents left unlabelled when learners have a 15% error rate and a 10% error rate [13]. These results are summarized in Table 4. Note that we want learners to produce as few unlabelled documents as possible, so a low score in the table is preferable. The learners with the best scores are written in bold face.

13. Admittedly 15% and 10% are somewhat arbitrary choices.

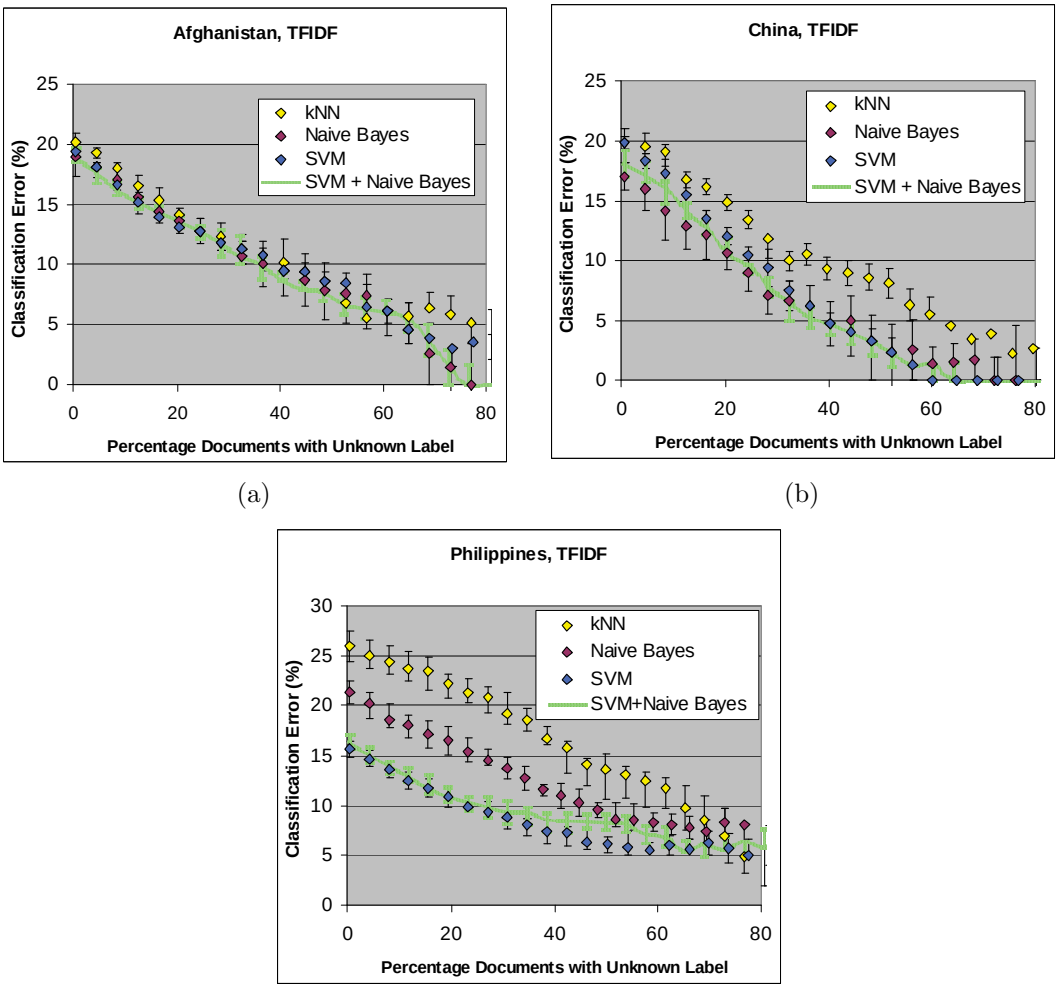


Figure 8: *Classification accuracy when classifier leaves certain documents unlabelled.*

No single learner in the table achieves the best score in every instance. The SVM and Naïve Bayes learners alternate with the best and second-best scores depending upon the corpus, while the kNN learner is always the worst, leaving the most documents unlabelled in every case. In an attempt to further improve on the learners' performance, we considered the possibility of *combining* the confidence scores of the learners—our belief was that cases may arise where one learner may have relatively low confidence in its categorization of a particular document, while the other learners may be more confident. The combined confidence scores are computed as

$$C(c,d) = \omega_{\mathrm{kNN}} \cdot C_{\mathrm{kNN}}(c,d) + \omega_{\mathrm{NB}} \cdot C_{\mathrm{NB}}(c,d) + \omega_{\mathrm{SVM}} \cdot C_{\mathrm{SVM}}(c,d) \quad (8)$$

$$C(\overline{c},d) = \omega_{\mathrm{kNN}} \cdot C_{\mathrm{kNN}}(\overline{c},d) + \omega_{\mathrm{NB}} \cdot C_{\mathrm{NB}}(\overline{c},d) + \omega_{\mathrm{SVM}} \cdot C_{\mathrm{SVM}}(\overline{c},d) \quad (9)$$

$$C(d) = \max\left(C(c,d), C(\overline{c},d)\right), \quad (10)$$

Table 4: *Percentage documents left unlabelled for learners' error rates*

		kNN	Naïve Bayes	SVM	SVM + NB
Afghanistan	Error Rate = 10%	41%	**37%**	38%	36%
	Error Rate = 15%	17%	14%	**13%**	12%
China	Error Rate = 10%	32%	**22%**	26%	23%
	Error Rate = 15%	21%	**6%**	13%	10%
Philippines	Error Rate = 10%	65%	45%	**23%**	26%
	Error Rate = 15%	44%	24%	**4%**	4%

where $C(c, d)$ is the combined confidence score that document d is in category c, $C_{\text{learner}}(c, d)$ is the confidence of the specified "learner" that document d is in category c, and ω_{learner} is the weight assigned to the specified learner. The larger of $C(c, d)$ and $C(\overline{c}, d)$ indicates the categorization decision of the combined learner, and $C(d)$ is the confidence of that decision.

We investigated the particular case[14] where $\omega_{\text{kNN}} = 0$, $\omega_{\text{NB}} = 0.5$, and $\omega_{\text{SVM}} = 0.5$. The performance of this case, as well as the three individual learners, is depicted in Figure 8 and tabulated in Table 4.

While the combined SVM + NB learner may not always be the best learner in every corpus, it is *always* better than the second-place learner. So, for example, Table 4 indicates that Naïve Bayes is the best choice for the China corpus; the combined SVM + NB learner outperforms the second-place SVM learner in this case. In the Philippines corpus, while SVM is the best choice, the combined SVM + NB learner is close and vastly outperforms the second-place Naïve Bayes learner. The lesson is that the combined learner offers a consistently reliable performance across corpora that is very competitive with the optimal choice. A final benefit of this learner is that it can provide some simple user feedback (in the form of the Naïve Bayes decision) to indicate the rationale behind a classification decision.

14. Our intuition for choosing this particular configuration of ω values was based on the fact that SVM and Naïve Bayes were always the top two learners, whereas kNN was always the worst. Since there was no clear winner between SVM and Naïve Bayes, we weighted them equally. Indeed, when we experimented with increasing the weight ω_{kNN} above zero, the results of the combined learner (not shown) were uniformly *poorer* than the case where $\omega_{\text{kNN}} = 0$. Choosing optimal weights, ω, for the three learners is a complex task and is beyond the scope of our work. We intended merely to demonstrate that by judicious combining of confidence intervals, it is possible to strike a balance such that the performance is preferable to any individual learner.

4.7 Experiment 7: Cross-domain learning

Thus far, our experiments have focused on the case of intra-domain or intra-corpus learning, whereby the documents used to *train* the machine learner are selected from the same corpus as the documents used to *test* the machine learner [15]. This is typical of the machine learning methodology—in fact, the entire premise of machine learning is based on the idea that a learner can be trained on examples that are representative of future test samples. However, we were interested in determining whether a classifier that learned a security classification rule based on one corpus could use that rule in assigning classifications to documents from a different corpus. We call this cross-domain or inter-domain learning.

Figure 9 displays the results of our investigation into cross-domain learning. In the figure, the x-axis indicates the corpus from which the test documents were selected. So, for instance, the three data points corresponding to the Philippines were generated by training a machine learner with a corpus created by combining the Afghanistan and China corpora and then testing the learner on the Philippines corpus. For this experiment, since the training and test corpora were *different*, it was not appropriate to use n-fold cross-validation to evaluate the learner's accuracy. A single train/test run involved creating a training set by taking a bootstrap sample from the training corpus (see [32] for a discussion on bootstrapping) and likewise creating a test set by taking a bootstrap sample from the test corpus. To add confidence intervals to our results, we once again used the percentile method of [33] as discussed in Section 4.1.

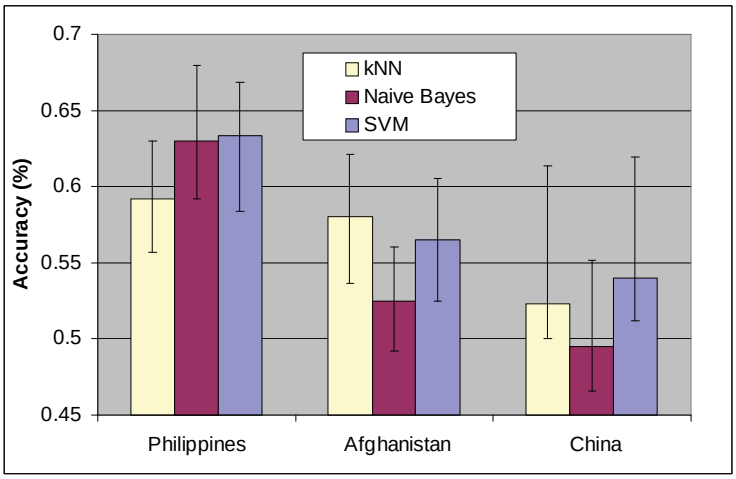


Figure 9: *Cross-domain classification performance: training set includes no documents from test domain.*

15. As is made clear in Section 3.2.2, the training set is always mutually exclusive from the test set by virtue of the 10-fold cross-validation. We are pointing out here simply that for intra-domain learning the training and test sets are drawn from the same corpus.

We observe from Figure 9 that the performance of the classifiers is very strongly influenced by the domain of the training set; in fact, the cross-domain performance of the security classifier is hardly better than could be achieved by guessing in many cases. This is not surprising, since we would expect that the more closely tied the test set is to the training set, the better our performance would be.

The major conclusion we can draw from the results of this experiment is that a practical implementation of a security classification system would require a multi-stage classifier, where the initial classification stage made a determination of the topic or *domain* of the document, and the subsequent stage selected the appropriate model for applying the security classification. Fortunately, finding the topic of a document is a much easier problem; a simple experiment demonstrated that we could accurately classify the domain of an incoming text (selecting among Afghanistan, China, and Philippines) with $98.7\% +/- 0.3\%$ accuracy using a Naïve Bayes classifier.

4.8 Experiment 8: The temporal effect

The final set of experiments we conducted was intended to investigate how well a learner would perform when classifying test documents created in an era different from the training set. Specifically, we were interested in a scenario where a machine learner is trained at a certain point in time, following which a dramatic change to security policy or political climate takes place. We wanted to know if the learner would still be able to operate accurately after this event. We refer to the influence of a change in era on a learner's performance as a "temporal effect".

We undertook two studies of the temporal effect using corpora from the Philippines and corpora from Cuba, as discussed below.

4.8.1 The Philippines experiment

For our first temporal experiment, we used two corpora from the Philippines from two different political eras. The first corpus was used to train the learner and consisted of the Philippines corpus from previous sections; documents in this corpus were created in the era from 1973 to 1981, which corresponds to the period during which Filipino President Ferdinand Marcos ruled by martial law. A second corpus was used to test the learner and consisted of Philippines documents created in the era from 1982 to 1986; this corresponds to the period during which Marcos lifted martial law, but continued ruling as president until his exile. Statistics on both Philippines corpora are provided in Table 5.

In all cases, the experiments were conducted using *tfidf* term weighting, unigram tokenization, no stemming, and no dimensionality reduction, as per the results of

Table 5: *Corpus Statistics for Temporal Study*

Corpus	# Docs	# Unclass	# Class
Philippines (1973-1981)	262	131	131
Philippines (1982-1986)	262	131	131
Pre-Cuban Missile Crisis	202	101	101
During Cuban Missile Crisis	236	118	118
Post-Cuban Missile Crisis	148	74	74

Sections 4.2, 4.3, and 4.4. To add statistical confidence measures to our results, we used the bootstrapping and percentile techniques discussed in Section 4.7; once again, n-fold cross validation was not applicable in this case since two different corpora were used for training and testing.

Figure 10 depicts the results of our temporal experiments on the Philippines corpora. For comparative purposes, results are provided for both the intra-period learning instance (where the training and test sets were both drawn from the 1982-1986 corpus) and the inter-period learning instance (where the training set is the 1973-1981 corpus).

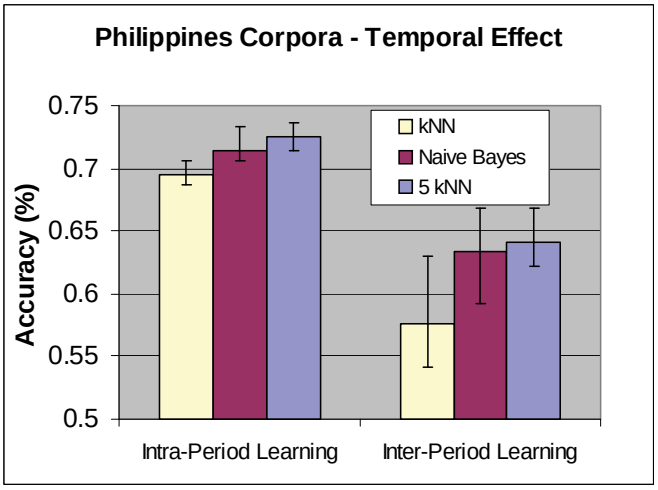


Figure 10: *Temporal effect for the Philippines: training corpus was from 1973-1981, test corpus was from 1982-1986.*

We observe a significant degradation in accuracy for all three learners in the inter-period case. This suggests that the classification rules learned from the temporally earlier training set do not apply when significant policy changes have taken place. Of note is that the kNN learner is most significantly affected by the change in era. This is possibly a result of the fact that kNN classifiers tend to be very sensitive to irrelevant features and often suffer performance degradation if irrelevant features are

not pruned out [41]. The inter-period scenario is bound to introduce many irrelevant features (i.e., irrelevant from the point of view of a learner trained on data from an earlier era), which lead the kNN classifier astray.

4.8.2 The Cuban Missile Crisis experiment

In our second temporal experiment, we studied three corpora (also obtained from the DNSA) related to the events surrounding the Cuban Missile Crisis. All corpora contain documents created during 1962, the year that the crisis took place. The first corpus consisted of documents from January 1 to October 15. Documents in this first corpus were created during the months prior to the crisis and these were the documents used to train the learner. The second corpus consisted of documents created during the period from October 16 to October 28, representing the so-called "Thirteen Days" during which the crisis took place. The third corpus contained documents from October 29 to December 31—these documents surround the aftermath of the event. The learner trained on the pre-crisis data was evaluated on both the corpus representing the actual crisis and the corpus representing the aftermath. Statistics on the three corpora are provided in Table 5.

In Figure 11, the results of the temporal experiments on the Cuban Missile Crisis corpora are reported. Interestingly, the SVM and Naïve Bayes learners show little loss in performance on the corpus containing the October 16 - October 28 data, representing the documents generated during the crisis itself. The kNN, however, has strongly degraded performance, again, due to the presence of irrelevant features.

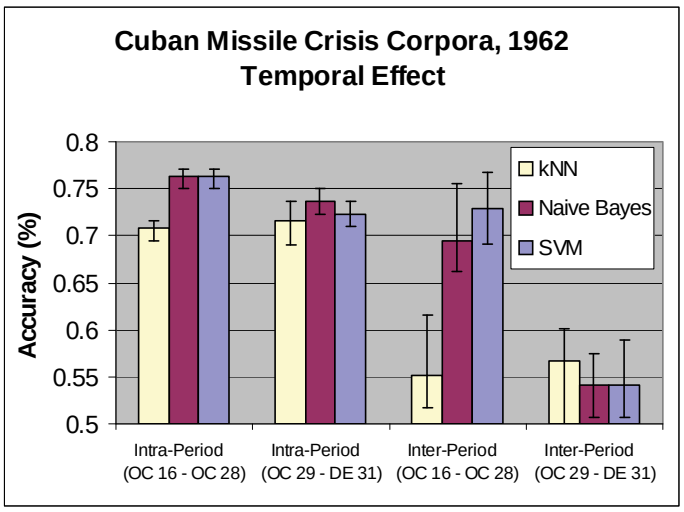


Figure 11: *Temporal effect for the Cuban Missile Crisis: training corpus was from January 1 - October 15, test corpora were from October 16 - October 28, and from October 29 - December 31.*

After the crisis is over, however, all of the classifiers perform very poorly. Upon further inspection of the results, we observed that the classifiers operating on the post-crisis corpus tended to strongly *overclassify* documents, assigning over 75% of documents to the "Classified" category. This is possibly a result of the fact that previously sensitive information might be less sensitive in a post-crisis world since the crisis precipitated public awareness of government policies and actions.

In general, our investigations of the temporal effect lead us to conclude that an automated security classification system must be flexible enough to adapt to changes to a learned classification rule, whether this be the result of new security policy due to a change in political climate or due to documents reaching their declassification date. An automated system would need to be retrained with user feedback, suggesting a Naïve Bayes classifier may be preferable to a Support Vector Machine classifier as per Section 4.5, which discussed the complexity of updating a learner's training model.

5 Recommendations and future direction

Section 4 presented the results of a series of experiments intended to evaluate the performance of traditional SNLP techniques when applied to document security classification. This section provides some general recommendations for system design based on the outcomes of our experiments, along with several potential avenues for exploitation, and directions for future research.

5.1 Recommendations for system design and optimization

In general, we showed that with appropriate tuning and component selection, it is possible to reliably obtain *intra-domain* classification accuracies near 80%. If classifiers are permitted to leave some data unlabelled, accuracies of higher than 90% are possible with appropriately tailored learners. The decisions made by the SNLP classifiers are remarkably accurate, considering these decisions are made purely on the basis of statistical pattern matching and do not incorporate any external contextual knowledge such as the document's author, date of authorship, network of origin, etc.

The following are our recommendations to tailor the SNLP model of Figure 1 to the task of automating security classification:

- **Tokenization**:Unigram tokenization is sufficient. Attempts to improve a classifier's performance beyond a bag-of-words baseline by retaining N-Grams were unsuccessful.
- **Stemming**: Stemming is neither beneficial nor harmful. Although stemming will result in a modest reduction in initial dimensionality (and thus a commensurate reduction in subsequent processing time) it does not improve classification performance. The initial processing burden of implementing stemming does not recommend it for our use.
- **Term Weighting**: For moderate dimensionality reduction (i.e., retaining more than 10% of features), binary term weighting and *tfidf* weights achieve comparable performance. As dimensionality reduction becomes more aggressive *tfidf* weights are recommended. Thus, *tfidf* weighting is never harmful and is beneficial in the lower dimensionality case.
- **Dimensionality Reduction**: The information gain and chi-square feature selection techniques are equally effective in reducing dimensionality. Either is recommended. Little performance degradation is observed when greater than 10% of features are retained. Below 10%, a sharp decline in performance is observed. For optimal performance we recommend retaining all features.
- **Machine Learner**: We recommend a classifier that linearly combines the confidence values of the SVM and Naïve Bayes learners and leaves "low confidence" test

documents as unlabelled (following the strategy described in Section 4.6). If forced to select only a single learner, we recommend the Naïve Bayes classifier based on the summary in Table 3.

Figure 12 shows a block diagram of an SNLP classifier implementing the above recommendations. This is a two-stage classifier as discussed in Section 4.7; the first stage is intended to determine the topical domain of the test document and the second stage selects the security classification based on the appropriately selected training model. The classifier decomposes the test document into unigrams, assigns a *tfidf* weight, selects the topical domain, and then uses the combined Naïve Bayes and SVM confidence levels to assign a classification to the document based on a confidence threshold; the classifier leaves open the possibility of not labelling the document if the confidence level is too low.

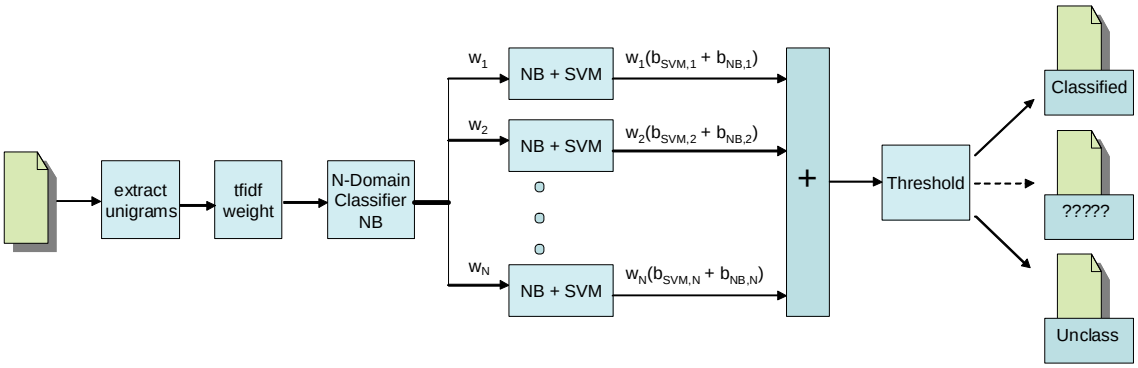


Figure 12: *Multi-domain statistical classifier*

In our experiments, we considered a simple 3-topic world in which the topics were either Afghanistan, China, or the Philippines. In this case, assigning a test document exclusively to a certain topic domain makes sense. A real-world scenario would have a much richer set of topics, however, where it is not hard to imagine that a document may be simultaneously related to a multitude of domains. For instance, a document concerning a NATO ally's acquisition of new artillery for overseas deployment could easily be described as a having a topic of "weaponry", "Afghanistan", or "NATO", or some combination of all three. While it is possible to limit the topic of a document to a single choice (in which case one of the weights w_i in Figure 12 would have a value of 1 and the rest would have values of 0), the classifier in Figure 12 leaves open the possibility that a document may be influenced by more than one topical domain where the proposed classifier would permit appropriate weights w_i to be assigned to each potential topic. Determining an appropriate set of topics into which a test document

can be sorted is a non-trivial problem and is beyond the scope of this paper; it is an area we plan to pursue in the future.

5.2 Ancillary benefits and exploitation possibilities

Although the primary impetus for our research is to facilitate secure information sharing, an automated security classification capability would have a number of additional applications. The following ancillary benefits should not be overlooked:

- **Sanitization and redaction**: Redaction—the act of removing sensitive information from a document—is frequently performed on government documents in order to downgrade their security classifications to make them suitable for release to individuals at lower levels of classification. A frequent application of redaction is in preparing documents for public release to service access to information requests (in compliance with the Access to Information Act). In Fiscal Year 2008-2009 alone, the Government of Canada spent over $48 million servicing access to information requests [42]. A tool that automates security classification could potentially be used to determine the "new" classification of a redacted document to assist a human operator in asserting that all sensitive information had indeed been removed. This could lead to faster sanitization of classified documents and may help streamline the access to information process. Granted, the system would need to significantly increase its accuracy beyond the current 80-90% in order to be relied upon not to leak sensitive information.
- **Data aggregation**: As discussed in [43], the aggregation of non-sensitive or less sensitive information presents a particularly difficult problem in information security. It is well known that certain information may have little or no sensitivity when taken in isolation, but when this "less sensitive" information is combined the aggregate may become highly sensitive. The data aggregation phenomenon is sometimes referred to a "mosaic" problem, where individual tiles of a mosaic convey little information, but the synthesis of many tiles reveals a clear picture. Manually and *efficiently* identifying which data aggregates constitute sensitive information is exceedingly challenging. An automated security classification process could possibly identify sensitive aggregates by treating a collection of individual documents as a single entity and then determining its classification (which may indeed be higher than the classification of any individual document in the collection). Identifying sensitive aggregates is an important first step towards solving the problem of data leakage through inference.
- **Real-time labelling**: Not only would an automated system be capable of assisting in the classification of documents, it could also be used as an aid in monitoring the security level of real-time applications such as instant message chat or email. Individual emails could simply be treated in a similar fashion as documents. Chat sessions could be continuously monitored and a user could be alerted if the content of a particular session was too sensitive for one of the participants or for the

particular network on which the session was taking place.

- **Finding data leakage**: By proactively labelling (i.e., through an automated process) all files, emails, and chat in an enterprise, the egress of data can be monitored and/or blocked at network exit points. This is in-line with the philosophy of current data leakage prevention tools, which actively track labelled data within the enterprise and allow for the identification of architectural weaknesses. An automated system could make use of deep content inspection tools (e.g., Purifile) that would allow the algorithms to detect covert data such as white-on-white text, very small fonts, appended metadata, and hidden edits.
- **Identifying opportunities to share**: Classified documents may be additionally restricted to be releasable only to a particular COI; it is quite possible that these documents would be valuable to individuals in other COIs to which they have not been released. An automated system could examine repositories of classified documents and identify those COIs to which the documents may be of interest and suggest opportunities to share information.

5.3 Future direction

The intent of this current research was to determine the extent to which machine learning could be relied upon to automate the process of security classification. Although our experiments were conducted under somewhat idealized conditions [16], we achieved accuracies of between 80% to 90% with relatively small training sets. Ultimately we want to refine our system, increasing accuracy and robustness to the point where it could be used by DND or the Canadian Forces (CF) to assist in classifying unstructured text, including documents, email and chat sessions. We intend to pursue the following research in order to further this ultimate goal:

- **Incorporate context into decisions**: Our current model focuses purely on SNLP techniques where only the words in the document are examined to reach a classification decision. Clearly, most unstructured text does not exist in such vacuum-like conditions and the classification could be based in part on additional knowledge, including the document's author, the date the document was composed, the classification of any previously existing versions, the recipient list (in the case of email or instant message traffic), the results of a standard dirty-word search, and the network where the document originated. We refer to these external elements of knowledge as "context"; it is an open problem how best to incorporate existing context into our model's decision-making process.
- **Adaptability**: As discussed in Section 4.8, learned models become obsolete as time passes, policy changes, documents reach their declassification date, or data is overcome by events and is no longer relevant. A useful classification system must retrain the model on a regular basis to account for this model drift, placing a

16. i.e., limited corpus domains and binary decisions

greater emphasis on new training examples to ensure new policies are reflected in the training data. Determining how often the model needs to be retrained and how much new data is required to maintain a stable level of performance are important questions for future research.

- **Multi-domain identification**: While Section 5.1 suggests a two-stage classifier where the initial stage identifies the topical domain of a document, there is no guidance provided regarding how to determine relevant topical domains for a collection of real-world documents. Ultimately the job of the initial stage is to sort test documents according to a pre-existing high-level taxonomy. Using existing military taxonomies, or automatically generated application-specific taxonomies (see a discussion of taxonomies in [16]) are possible options. Choosing the taxonomy for the initial stage classifier is an important open question.
- **Human accuracy**: We reach accuracies near 80% for simple binary decisions (classified or unclassified), and exceed 90% when we allow for the possibility of "uncertain" decisions. While the prospect of underclassifying 10% - 20% of sensitive documents is unsettling, it is of significant interest to learn how accurate a human being would be in classifying the same documents. Considering that some estimates place the percentage of overclassified documents in the U.S. DoD at 50%, clearly human beings are not as accurate as we might imagine [17]. A careful survey of human classification would be invaluable in assessing the merits of an automated system and would give insight into the reliability of the training sets used to train our models.
- **Detailed design**: Ultimately, for DND to leverage an automated classification capability, we require a detailed design describing how to implement automated classification as a service that could easily plug-in to existing architectures.

17. Note that many overclassified documents are not necessarily overclassified "by mistake", but rather may have been overclassified to expedite working with the documents on a sensitive network.

6 Conclusion

This paper detailed the results of a series of experiments evaluating the application of statistical natural language processing (SNLP) techniques to the problem of automated security classification. Using simple text processing techniques on a collection of declassified government documents, we were able to achieve classification accuracies greater than 80%; using a more sophisticated approach that combined Bayesian and Support Vector Machine classifiers, our accuracy improved to over 90% (bearing in mind that a subset of documents was left unlabelled in this case). We demonstrated the strong dependence of the security classification task on the topical domain of the documents, concluding that a multi-stage classifier is recommended in practice whereby an initial stage determines the topic of a document and a second stage selects an appropriate set of models for security classification. Additionally, we showed that changes in policy or political era cause machine learners to lose fidelity as their training data loses relevance; this suggests the need for periodic re-training with more representative data.

The performance of our classifier is encouraging and strongly suggests that SNLP should be a major component of any future implementation of an automated security classifier. By combining our methods with existing techniques—such as dirty word search—and by introducing contextual information—such as document authorship—we believe accuracy can be further improved and we are actively pursuing these solutions.

References

[1] Cavas, Christopher (2010), Petraeus: US must share more info with allies (online), http://www.defensenews.com/story.php?i=4623591 (Access Date: July 2, 2010).

[2] U.S. Government Printing Office (2004), Too Many Secrets: Overclassification as a Barrier to Critical Information Sharing. Hearing Before the Committee on Government Reforms, US House of Representatives, p. 82.

[3] Carlstrom, Gregg (2010), Connecting agencies: Can Obama end mistrust in intel community? (online), http://www.federaltimes.com/article/20100110/AGENCY04/1100308/-1/RSS (Access Date: July 8, 2010).

[4] Digital National Security Archive (online), http://nsarchive.chadwyck.com/home.do (Access Date: September 8, 2009).

[5] Meiler, Peter-Paul and Schmeing, Michael (2009), Secure Service Oriented Architectures (SOA) supporting NEC, (Technical Report TR-IST-061) NATO.

[6] Wolf, Ulrich (2006), Does NATO meet the challenge of the information era?, In *23rd International Workshop on Global Security*, Berlin, Germany.

[7] Brown, Nancy (2003), Statement for the Record Before the 108th Congress Committee on Armed Services, US House of Representatives.

[8] McDonald, Andrew and Hepworth, Eleanor (2009), Information superiority: Enabling the need to share, Roke Manor Research Ltd. (online), http://www.roke.co.uk/resources/white-papers/information-superiority-enabling-the-need-to-share.pdf.

[9] Grandison, T., Bilger, M., O'Connor, L., Graf, M., Swimmer, M., Schunter, M., Wespi, A., and Zunic, N. (2007), Elevating the Discussion on Security Management: The Data Centric Paradigm, In *2nd IEEE/IFIP International Workshop on Business-Driven IT Management*.

[10] Government of Canada Security Policy (online), http://www.tbs-sct.gc.ca/pol/doc-eng.aspx?id=12322\&section=text (Access Date: August 26, 2009).

[11] Department of National Defence (2004), Metadata Application Profile: Unstructured Information Resources, (Technical Report IM Standard 456.21) DND.

[12] Magar, Alan (2005), Investigation of Technologies and Techniques for Labelling Information Objects to Support Access Management, (DRDC-Ottawa CR 2005-166) Defence R&D Canada – Ottawa.

[13] Brown, David (2010), Developing an automatic document classification system: A review of current literature and future directions, (DRDC-Ottawa TM 2009-269) Defence R&D Canada – Ottawa.

[14] Clark, K. (2008), Automated Security Classification, Master's thesis, Vrije Universiteit.

[15] Mathkour, H., Touir, A., and Al-Sanie, W. (2004), Automatic Information Classifier Using Rhetorical Structure Theory, In *IIWAS*, Jakarta.

[16] Magar, Alan (2010), Automating Security Classification using Commercial Content Analysis Products, (To be published) Defence R&D Canada – Ottawa.

[17] Thomas, Scott, Collaboration without boundaries: Cross-domain solutions for network-centric operations (online), `http://tridsys.com/Collaboration\_Gateway\_White\_Paper.pdf` (Access Date: September 7, 2010).

[18] Fletcher, Boyd (2006), XMPP and Cross Domain Collaborative Information Environment (CDCIE) Overview, In *SensorNet summer workshop on net-ready sensors: the way forward*, Oak Ridge, Tennessee.

[19] Ames, Scott (2009), SMART.NeXt capabilities briefing, In *3rd Annual UCDMO Conference 2009*, Adelphi, Maryland.

[20] Rooney, Thomas (2009), eXMeritus HardwareWall: Secure Data Transfer System, In *3rd Annual UCDMO Conference 2009*, Adelphi, Maryland.

[21] Sebastiani, Fabrizio (2002), Machine learning in automated text categorization, *ACM Computing Surveys*, 34(1), 1–47.

[22] Porter, M. (1997), Readings in Information Retrieval, An Algorithm for Suffix Stripping, pp. 313–316, Morgan Kaufman Publishers.

[23] Yang, Y. and Pedersen, J. (1997), A Comparative Study on Feature Selection in Text Categorization, In *Proceedings of ICML-97, 14th International Conference on Machine Learning*, Nashville.

[24] Rogati, M. and Yang, Y. (2002), High-Performing Feature Selection for Text Classification, In *Proceedings of the 11th international conference on Information and Knowledge Management.*

[25] Manning, C., Raghavan, P., and Schutze, H. (2009), An Introduction to Information Retrieval, Cambridge University Press.

[26] Bennett, K. and Campbell, C. (2000), Support Vector Machines: Hype or Hallelujah?, *SIGKDD Explorations*, 2, 1–13.

[27] Fletcher, T., Support Vector Machines Explained (online), `http://www.cs.ucl.ac.uk/staff/T.Fletcher` (Access Date: September 3, 2009).

[28] Joachims, T. (1998), Text Categorization with Support Vector Machines: learning with many relevant features, In *In Proc. of ECML-98, 10th European Conference on Machine Learning*, pp. 137–142, Heidelberg.

[29] Deerwester, S., Dumais, S., Furnas, G., Landauer, T., and Harshman, R. (1990), Indexing by Latent Semantic Analysis, *Journal of the American Society for Information Science*, 41(6), 391–407.

[30] Lan, M., Sung, S., Low, H., and Tan, C. (2005), A Comparative Study on Term Weighting Schemes for Text Categorization, In *In Proc. of International Joint Conference on Neural Networks*, Montreal, Canada.

[31] Yu, Bei (2006), An Evaluation of Text Classification Methods for Literary Study, Ph.D. thesis, University of Illinois at Urbana-Champaign.

[32] Kohavi, Ron (1995), Wrappers for Performance Enhancement and Oblivious Decision Graphs, Ph.D. thesis, Stanford University.

[33] Efron, B. and Tibshirani, R. (1993), An Introduction to the Bootstrap, Chapman & Hall.

[34] Pang, B., Lee, L., and Vaithyanathan, S. (2002), Thumbs Up? Sentiment Classification Using Machine Learning Techniques, In *Proceedings of EMNLP*.

[35] Pomikalek, J. and Rehurek, R. (2007), The Influence of Preprocessing Parameters on Text Categorization, In *In Proc. of World Academy of Science, Engineering and Technology*, pp. 54–57, London, UK.

[36] Dumais, S., Platt, J., Heckerman, D., and Sahami, M. (1998), Inductive Learning Algorithms and Representations for Text Categorization, In *In Proc. of CIKM-98, 7th ACM Internation Conference on Information and Knowledge Management*, pp. 148–155, Washington.

[37] Aue, A. and Gamon, M. (2005), Customizing Sentiment Classifiers to New Domains: A Case Study, In *In Proc. RANLP, Recent Advances in Natural Language Processing*, Bulgaria.

[38] Fine, S. and Scheinberg, K. (2001), Efficient SVM Training Using Low-Rank Kernal Representations, *Journal of Machine Learning Research*, 2, 243–264.

[39] Kononenko, Igor (1993), Inductive and Bayesian Learning in Medical Diagnosis, *Applied Artificial Intelligence*, 7(4), 317–337.

[40] Kotsiantis, S. B. (2007), Supervised Machine Learning: A Review of Classification Techniques, *Informatica*, 31, 249–268.

[41] Yavuz, T. and Guvenir, H. (1998), Application of k-Nearest Neighbor or Feature Projections Classifier to Text Categorization, In *In Proc. of 13th International Symposium on Computer and Information Sciences*, pp. 135–142, Antalya, Turkey.

[42] Access to Information and Privacy Act: Bulletin Number 32B Statistical Reporting (online), `http://www.infosource.gc.ca/bulletin/2009/b/bulletin32b/bulletin32b00-eng.asp` (Access Date: July 5, 2010).

[43] NIST (2008), Guide for Mapping Types of Information and Information Systems to Security Categories, Vol. I, (Technical Report NIST-800-60).

Acronyms and abbreviations

CF Canadian Forces

COTS commercial off-the-shelf

COI community of interest

DBAC Direction Based Access Control

DLP Data Leakage Prevention

DND Department of National Defence

DNSA Digital National Security Archive

DoD Department of Defence

GSP Government of Canada Security Policy

IP internet protocol

ISSE Information Support Server Environment

k**NN** k-Nearest Neighbour

NATO North Atlantic Treaty Organization

NB Naïve Bayes

OHSUMED Oregon Health Sciences University Medicine

SNLP statistical natural language processing

SVM Support Vector Machine

tfidf term frequency—inverse document frequency

UCDMO Unified Cross Domain Management Office

This page intentionally left blank.

Annex A: Additional results

A.1 k-nearest neighbour

For the k-Nearest Neighbour learner, we used $k = 5$ to generate all the results reported in Section 4. The choice of $k = 5$ was based on some preliminary work shown in Figure A.1, which plots the accuracy of k-Nearest Neighbour learners as a function of k for each of our corpora, investigating the range $k \in (5, 10, 15, 20, 25, 30)$. We observe in all cases that a choice of $k = 5$ is reasonable, with the learner performing almost as well (or better) as a learner using larger values of k. In addition, a smaller choice of k reduces the computational complexity of classifying a new document, making $k = 5$ a desirable choice.

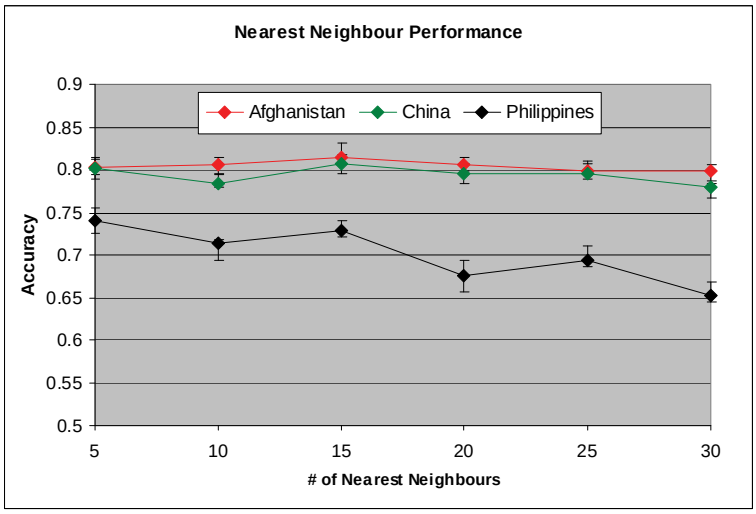


Figure A.1: *Effect of k on the accuracy of k-Nearest Neighbour learners.*

A.2 All feature selection results

This annex displays the effects of feature selection on the Afghanistan and China corpora to complement the results shown in Section 4.2.

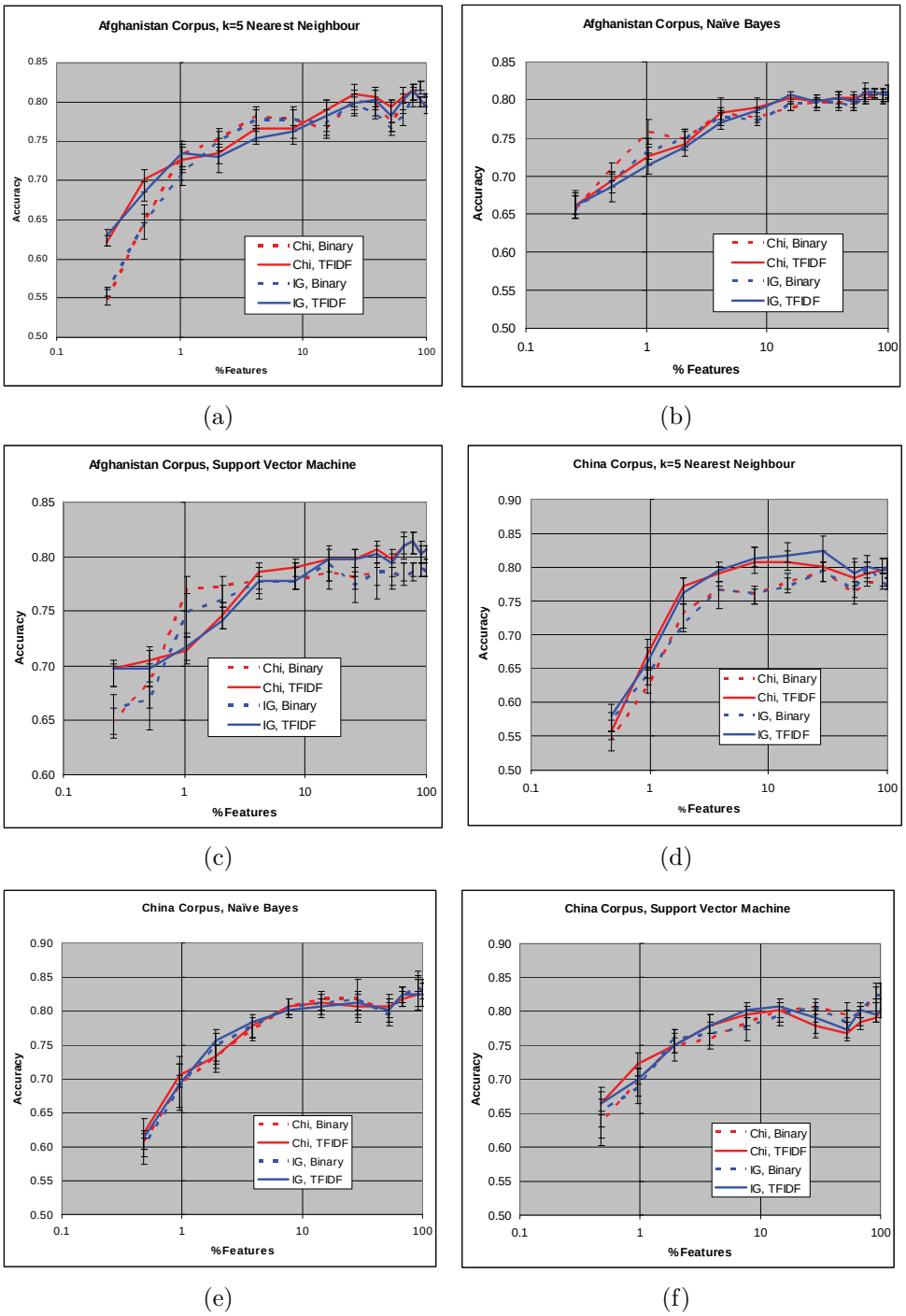


Figure A.2: *Classification accuracy of documents from the Afghanistan and China corpora.*

DOCUMENT CONTROL DATA

(Security classification of title, body of abstract and indexing annotation must be entered when document is classified)

1. ORIGINATOR (The name and address of the organization preparing the document. Organizations for whom the document was prepared, e.g. Centre sponsoring a contractor's report, or tasking agency, are entered in section 8.)

   Defence R&D Canada – Ottawa
   3701 Carling Avenue, Ottawa, Ontario, Canada
   K1A 0Z4

2. SECURITY CLASSIFICATION (Overall security classification of the document including special warning terms if applicable.)

   UNCLASSIFIED

3. TITLE (The complete document title as indicated on the title page. Its classification should be indicated by the appropriate abbreviation (S, C or U) in parentheses after the title.)

   Security classification using automated learning (SCALE)

4. AUTHORS (Last name, followed by initials – ranks, titles, etc. not to be used.)

   Brown, J.D.; Charlebois, D.

5. DATE OF PUBLICATION (Month and year of publication of document.)

   December 2010

6a. NO. OF PAGES (Total containing information. Include Annexes, Appendices, etc.)

   64

6b. NO. OF REFS (Total cited in document.)

   43

7. DESCRIPTIVE NOTES (The category of the document, e.g. technical report, technical note or memorandum. If appropriate, enter the type of report, e.g. interim, progress, summary, annual or final. Give the inclusive dates when a specific reporting period is covered.)

   Technical Memorandum

8. SPONSORING ACTIVITY (The name of the department project office or laboratory sponsoring the research and development – include address.)

   Defence R&D Canada – Ottawa
   3701 Carling Avenue, Ottawa, Ontario, Canada K1A 0Z4

9a. PROJECT NO. (The applicable research and development project number under which the document was written. Please specify whether project or grant.)

   15bc05

9b. GRANT OR CONTRACT NO. (If appropriate, the applicable number under which the document was written.)

10a. ORIGINATOR'S DOCUMENT NUMBER (The official document number by which the document is identified by the originating activity. This number must be unique to this document.)

   DRDC Ottawa TM 2010-215

10b. OTHER DOCUMENT NO(s). (Any other numbers which may be assigned this document either by the originator or by the sponsor.)

11. DOCUMENT AVAILABILITY (Any limitations on further dissemination of the document, other than those imposed by security classification.)

   (X) Unlimited distribution
   ( ) Defence departments and defence contractors; further distribution only as approved
   ( ) Defence departments and Canadian defence contractors; further distribution only as approved
   ( ) Government departments and agencies; further distribution only as approved
   ( ) Defence departments; further distribution only as approved
   ( ) Other (please specify):

12. DOCUMENT ANNOUNCEMENT (Any limitation to the bibliographic announcement of this document. This will normally correspond to the Document Availability (11). However, where further distribution (beyond the audience specified in (11)) is possible, a wider announcement audience may be selected.)

13. ABSTRACT (A brief and factual summary of the document. It may also appear elsewhere in the body of the document itself. It is highly desirable that the abstract of classified documents be unclassified. Each paragraph of the abstract shall begin with an indication of the security classification of the information in the paragraph (unless the document itself is unclassified) represented as (S), (C), (R), or (U). It is not necessary to include here abstracts in both official languages unless the text is bilingual.)

Automating the process of assigning security classifications to unstructured text would facilitate a transition to a data-centric architecture—one that promotes information sharing, in which all data in an organization are electronically labelled. In this document, we report the results of a series of experiments conducted to investigate the effectiveness of using statistical natural language processing and machine learning techniques to automatically assign security classifications to documents. We present guidelines for selecting parameters to maximize the accuracy of a machine learning algorithm's classification decisions for several well-defined collections of documents. We examine the significance of a document's topic and the effect of security policy changes on the ability of our system to automate classification; we include design recommendations to address both topic and policy considerations. Our classification techniques prove effective at assessing a document's sensitivity, achieving accuracies upwards of 80%.

14. KEYWORDS, DESCRIPTORS or IDENTIFIERS (Technically meaningful terms or short phrases that characterize a document and could be helpful in cataloguing the document. They should be selected so that no security classification is required. Identifiers, such as equipment model designation, trade name, military project code name, geographic location may also be included. If possible keywords should be selected from a published thesaurus. e.g. Thesaurus of Engineering and Scientific Terms (TEST) and that thesaurus identified. If it is not possible to select indexing terms which are Unclassified, the classification of each should be indicated as with the title.)

security classification
text categorization
trusted labelling
statistical natural language processing
cross-domain solutions

Defence R&D Canada

Canada's leader in Defence
and National Security
Science and Technology

R & D pour la défense Canada

Chef de file au Canada en matière
de science et de technologie pour
la défense et la sécurité nationale

www.drdc-rddc.gc.ca